Hash-Collecting System : Applying Freenet Routing Method to Named Data Networking

Natsuko Fukuda
School of Fundamental Science and Engineering
Waseda University
Tokyo, Japan
kinokonoyama73@asagi.waseda.jp

Hidenori Nakazato
School of Fundamental Science and Engineering
Waseda University
Tokyo, Japan
nakazato@waseda.jp

***Abstract*—As network usage shifts to content retrieval and its traffic increases, Information-Centric Networking (ICN) is collecting interests. The purpose of this paper is to improve the efficiency of content retrieval. Our proposing *Hash-Collecting System* applies Freenet routing method to Named Data Networking (NDN) forwarding system. The average number of hops required to retrieve contents is reduced by our proposing method from the standard NDN forwarding system. The proposed method enables finding content in near-by node from the requesting party and can reduce the content retrieval traffic.**

***Index Terms*—ICN, NDN, Freenet, Information-Centric Networking, Named Data Networking**

I. Introduction

As the Internet usage shifts from host access to content retrieval, Information-Centric Networking (ICN) is collecting interest. ICN uses content name for forwarding packets while current IP network uses IP address [1]. On ICN, routers can cache contents and users can get a content from a nearer node than the server of the content. In this way, ICN is expected to contribute to load distribution and delay reduction.

Named Data Networking (NDN) is one implementation of ICN [2]. In this paper, we incorporate the routing method of Freenet [3] to NDN forwarding system. The purpose of this paper is to improve the efficiency of NDN content retrieval exploiting the idea of Freenet.

The rest of the paper is organized as follows. Section II gives overview of NDN and Freenet as well as existing research to improve content retrieval. Proposed mechanisms are given in Section III. Section IV evaluates the performance of the proposed mechanism and Section V gives conclusions.

II. Related Study

A. Named Data Networking (NDN)

The overall flow of NDN forwarding is shown in Fig.1. An *interest packet* is a packet which requests a content. This packet is sent from a *consumer* and carries the name of the requested content. A consumer is a node requesting contents. A *data packet* is a packet which transports the requested content from a *producer* or *Content Store.* A producer is a node serving the content. Content Store provides cache function which NDN routers are equipped with. A data packet is forwarded back on the same path as its corresponding interest packet in reverse order.

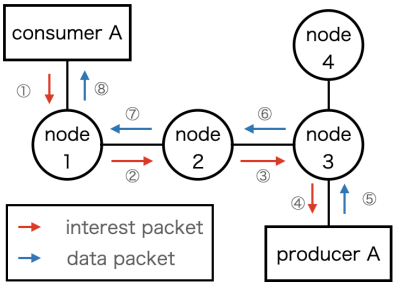

Fig. 1. Packet forwarding in NDN

An NDN router uses three data structures: Forwarding Information Base (FIB), Pending Interest Table (PIT), and Content Store (CS) for packet forwarding.

FIB is used to decide where to forward interest packets. FIB stores pairs of the prefix of content names and the *face ID* corresponding to the prefix. A face is an extended concept of network interface, and it can handle not only forwarding of packets over hardware network interfaces but also exchanging of packets directly with applications inside machines [1]. A face ID is the ID allotted to each face.

PIT is used for sending data packets back to the requesting consumers. PIT records the pairs of the content name of an interest packet and the faceID of the face at which the interest packet has arrived. When a data packet arrives at a router, the router refers PIT and send the data packet to the recorded face.

CS is a data structure to cache contents when data packets arrive. If an interest packet arrives at a router is requesting a content held in its CS, that content in the cache is sent back as a data packet without further forwarding the interest packet.

Fig. 2 shows the processing flow of interest packets. Suppose a consumer requests a content. Then, the consumer sends an interest packet with the name of the requested content. When the interest packet is received by a node, if the node is a producer which has the requested content, the producer creates a data packet with the requested content and sends back

This work was supported by JSPS KAKENHI Grant no. 19K11952 and MIC SCOPE Grant no. JPJ000595.

the data packet towards the consumer. If the node received the interest packet is not a producer but a router, the router confirms if the requested content is cached in its CS. If it is cached, the cached content is returned as a data packet and the interest packet is discarded.

If the requested content is not cached in its CS, the router examines its PIT. If the content name of the interest packet is found in the PIT, it means that other interest packets also seeking the same content were forwarded through this router recently and the requests were not yet satisfied. In that case, the router append the face ID from which the interest packet being processed is received to the existing PIT entry and the interest packet is discarded. If the requested content name is not recorded in the PIT, a new PIT entry is created and the content name and the face ID is recorded there.

Finally, the router refers FIB, looks for the prefix which matches longest with the name of the interest packet, and forwards the interest packet to the face matched in the FIB.

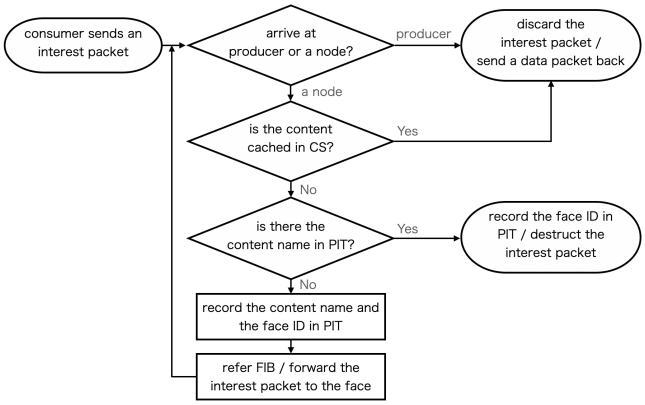

Fig. 2. The processing flow of interest packets

Fig.3 shows the processing flow of data packets. If the requested content is found in a CS or if an interest packet arrives at the producer of the content, a data packet is created with the requested content in it. When the data packet is received by a node, if the node is the consumer requesting the content, the forwarding ends. If the node is a router, the data packet is cached in case the content satisfies the cache policy, and then forwarded to the face recorded in its PIT. It means that the data packet with the requested content is sent back to the directions which the interest packets came from.

Fetching the Nearest Replica (FNR) [4] is a mechanism proposed to find contents in nearby nodes. In FNR, when a consumer requests a popular content, the content is fetched from the nearest replica regardless of whether it is on the best-route. In FNR, each router announces the popular content and each consumer fetch the content directly from CSs with redirection. The total traffic, average latency, and average cost are improved by FNR. FNR however requires a centralized coordinator to find the nearest replica. In Vicinity-based Replica Finding in NDN [5], the router has a *Content List* that contains the name prefixes from the CS and the list is used to advertise the availability of the contents and replicas from a node to other nodes in a vicinity. The interest packet is forwarded to the face written in the Content List if the requesting content is in the list. If the requested content is at a node a few hops away from the nodes exchanging the list, there is no clue in where to forward the interest packet.

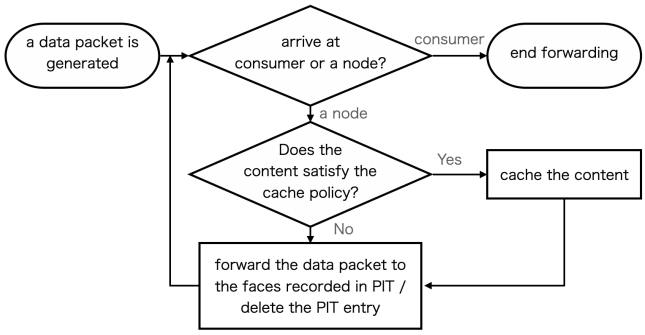

Fig. 3. The processing flow of data packets

B. Freenet

Freenet is one of peer-to-peer (P2P) network applications and it permits publication, replication, and retrieval of files while protecting the anonymity of the users [3]. In Freenet, forwarding or retrieval of data is conducted by using hash values. A node in Freenet has a routing table composed of pairs of a binary file key and a node address. This binary file key is obtained by applying a hash function to the file name. Since a "node" in Freenet is not a router but a computer connected at the end of a network unlike a node in NDN, the node has a node address, or an IP address. A Freenet user who requests a file has to calculate the binary file key of the file and sends it as a request to the user's own node. Each request is given a hops-to-live value, the limit of the number of hops of forwarding. The request is forwarded to the node address which has the closest binary file key in the routing table to the binary file key of the requesting file while hops-to-live value is set to non-zero value. When the found file is sent back to the user, the copies of the file are cached in each node. As more files circulate in the network, the efficiency of file retrieval is improved because the files whose hash values are closer are placed to closer nodes.

III. Proposal

A. Overview

We propose a mechanism called *Hash-Collecting System* that provides off-path caching function together with routing to the cached contents to be used with NDN. NDN with Hash-Collecting System combines NDN forwarding system with the caching/forwarding concept of Freenet.

NDN with Hash-Collecting System forwards interest packets in two phases: *Normal-Routing Strategy* phase and *Hash-Routing Strategy* phase. *Strategy* is a mechanism to decide the direction to forward packets in NDN term. We call interest packet forwarding using the standard NDN routing strategy using FIB, Normal-Routing Strategy. In addition, we propose

a new routing strategy, Hash-Routing Strategy. Hash-Routing Strategy forwards interest packets applying Freenet routing method. Interest packets are first forwarded by Hash-Routing Strategy within a hop limit specified by a parameter TTL and if no matching content is found, then the strategy is switched to Normal-Routing Strategy. In Hash-Routing Strategy, interest packets are forwarded using *Cache Table* (CT) in each router. CT holds cached contents like CS in NDN, and in addition, holds hash values of each content name and face IDs to which interest packets are to be forwarded (TABLE I).

TABLE I
STRUCTURE OF CT.

content name	content	hash value	Face ID
/prefix1/aaa/bbb	01001011· · ·	0d6477ad9fc16e4 debdee3ff72· · ·	2
/prefix2/ccc/ddd	00111001· · ·	2ad21cd6cfb385b c836dbc21a09· · ·	3
· · ·	· · ·	· · ·	· · ·

Additionally, in order to populate CT, Hash-Collecting System defines *Hash-Collector*. A Hash-Collector is a kind of consumer who helps to collect the contents whose hash values are "near" to its own *Collector ID*. To collect the contents, a Hash-Collector generates a new type of interest packet called *hash-interest packet*. A hash-interest packet carries a Collector ID instead of a content name. When a router receives a hash-interest packet, a process called *Collection-Judgement* is executed. In Collection-Judgement, the router searches its CT for the contents whose hash values are "near" to the Collector ID carried by the hash-interest packet. If near contents are found, one of the content is sent back to the Hash-Collector as a *hash-data packet* and if not, the hash-interest packet is forwarded by Hash-Routing Strategy until its TTL expires.

B. Interest/Data Packet Forwarding

Interest and data packets are forwarded as shown in Fig.4. The forwarding process of interest packet from a consumer is divided into two phases as mentioned before. In the first phase, the interest packet is forwarded with Hash-Routing Strategy. In this phase, the interest packet can be forwarded only in a limited number of hops (TTL). If the content is not found within the hop limit, a failure is conveyed to the router where the Hash-Routing Strategy is initiated by a *nack* packet, and the forwarding process goes to the second phase where the interest packet is forwarded with Normal-Routing Strategy. In either phase, once the requested content is found on the route, the data packet is sent back.

When an interest packet is received by a node, which is a router, in Hash-Routing Strategy, the router firstly checks if the requested content is cached in its CT, instead of CS, and records the content name and the face ID in PIT. Fig.5 shows the processing flow of interest packets in Hash-Routing Strategy after recording the necessary information in PIT. First, The router checks if CT is empty. If so, the interest packet is sent to a randomly selected face. If CT is not empty, the router

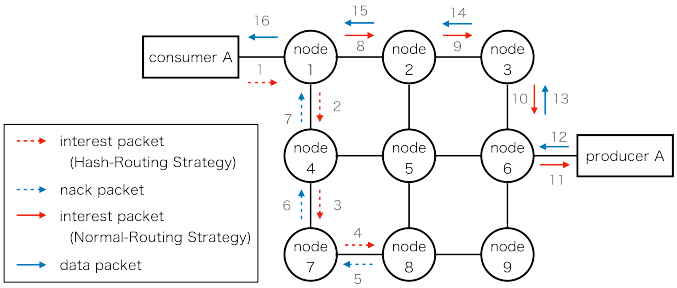

Fig. 4. Interest and Data Packet Forwarding

calculate the hash value of the content name of the interest packet by applying the same hash function as used in CT to the string of that content name. Then, the router examines CT and find the content which has the "nearest" hash value to the one calculated from the content name in the interest packet. The definition of "nearest" is that the absolute value of the difference between these two hash values is least. Finally, the router forwards the interest packet to the face specified in the nearest CT entry.

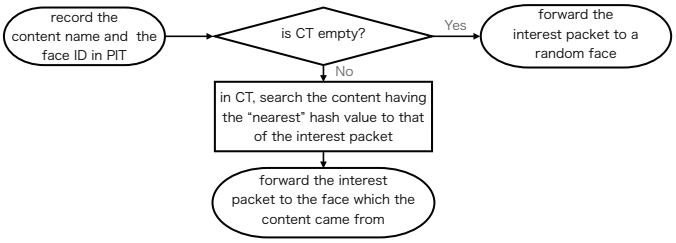

Fig. 5. Interest packet forwarding in Hash-Routing Strategy

C. Populating Cache Table

In order to populate CT, Hash-Collector System uses hash-interest and hash-data packets, and also Hash-Collector.

Hash-Collectors are deployed at some of the routers in the network. Each Hash-Collector is assigned an ID called Collector ID and the hash-interest packet sent from a Hash-Collector carries its Collector ID. The value of Collector ID is assigned from the same range as the hash values generated by the hash function used in Hash-Routing Strategy. How to allocate a Collector ID to each Hash-Collectors is left for future research. A hash-interest packet is forwarded within a TTL set by a Hash-Collector and if no content is found within the TTL, a nack packet is returned.

Forwarding of hash-interest and hash-data packets is shown in Fig.6. A hash-interest packet sent by a Hash-Collector seeks the content which has a hash value "near" the Collector ID of the Hash-Collector. The distance between two hash values or between a hash value and a Collector ID is defined as the absolute value of the difference of the two values. A distance is "near" when the distance is less than a certain threshold. The threshold is a parameter in our experiments. Please note the difference between "near" and "nearest" defined in section

III-B. “Near” is determined in terms of a threshold while “nearest” means the smallest distance.

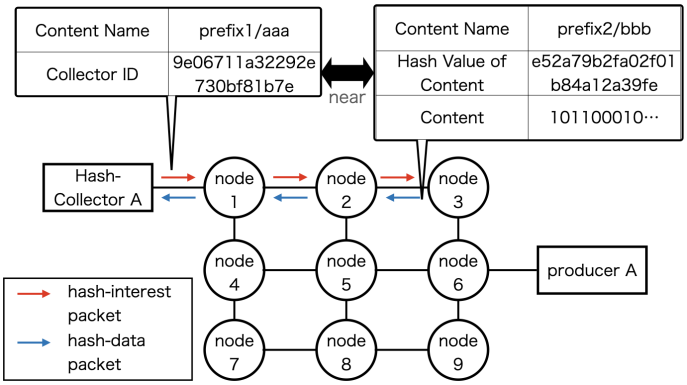

Fig. 6. The overall flow of packets related to Hash-Collector.

Upon reception of a hash-interest packet, the router checks if CT has any content which has its hash value near to the Collector ID carried by the hash-interest packet. We name these processes “Collection-Judgement.” Fig. 7 shows the processing flow of Collection-Judgement. When a hash-interest packet from a Hash-Collector arrives at a node, the router examines its CT and extract all of the contents whose hash values are near to the hash value of the hash-interest packet. If there is no near content, the hash-interest packet is forwarded with the Hash-Routing Strategy. If multiple near contents exist, the router selects one content to return to the Hash-Collector from these contents in some way, such as by selecting randomly. The way of selecting a content from near contentsa is a parameter and will be explained in section IV-B.

After selecting one near content, the router decide whether to return the content following a certain policy. The policy must be set so that the probability of picking up the content is lower on the closer node to Hash-Collector who request that content. The reason why the policy is needed is that we expect Hash-Collectors to collect as many kinds of contents as possible.

If the content is decided to be returned, the router creates a hash-data packet. A hash-data packet encapsulate the data packet to be returned within it. A hash-data packet also carries the Collector ID contained in the corresponding hash-interest packet. The Collector ID is necessary to return to the Hash-Collector using PIT.

On the way back to the Hash-Collector, the intermediate routes may cache the content carried by the hash-data packet.

IV. Experiment

A. Experimental Environment

The experiment is conducted on ndnSIM, which is an NDN simulator based on NS-3 [6].

The topology used in this experiment is GEANT topology. GEANT is the data network providing interconnection among NRENs (National Research and Education Network) of European countries [7].

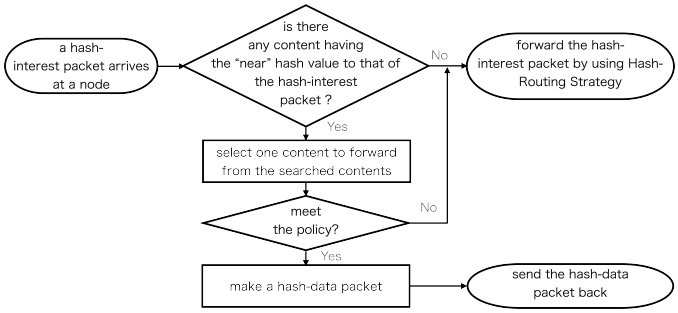

Fig. 7. The processing flow of Collection-Judgement.

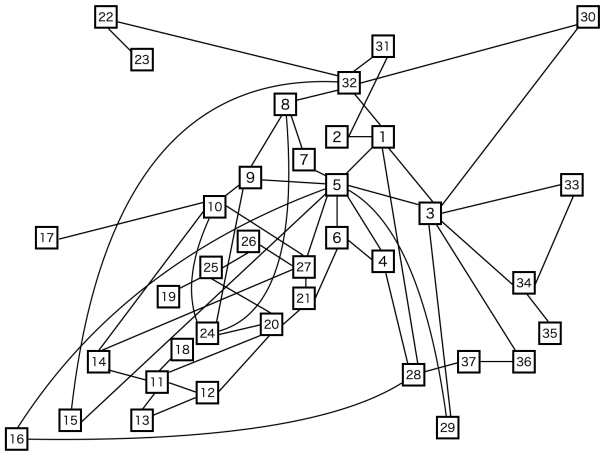

Fig. 8. GEANT topology.

In this experiment, the hash function used in Hash-Routing Strategy is the default hash function in C++, std::hash(). The possible value calculated by this function is from 0 to SIZE_MAX which is the max value of std::size_t.

The initial value of TTL in interest packet used in Hash-Routing Strategy and the one in hash-interest packet are both set to 4 in consumers and in Hash-Collectors, respectively.

B. Setting of Collection-Judgement

In Collection-Judgement, the router has to extract the contents whose hash value is near from CT. In this experiment, the threshold to determine the *near*-ness is set to 50%, 67%, 80% and 100% of SIZE_MAX. For example, when the threshold is fixed to 50% of SIZE_MAX, the router takes the contents whose hash value is in $\pm 25\%$ of SIZE_MAX away from Collector ID of the hash-interest packet.

If multiple contents are within the threshold, the way of selecting one content from these contents is as follows:

(a) Select the content whose hash value is nearest to that of the interest packet.
(b) Select one of the contents whose hash value is in the threshold randomly.

The policy alternatives deciding whether the content is sent back in Collection-Judgement is as follows:

(1) Content is sent back on all nodes except the one next to Hash-Collector who sent the hash-interest packet.

(2) Content is sent back with the probability p given by

$$p = 1 - < \text{\# hops from Hash-Collector} > \times 0.2$$

C. Placement of Consumers and Producers

In this experiment, multiple consumers are assumed and each consumer requests different prefixes. Let us call the consumer for observation *Main Consumer* (MC). In addition to MC, we place other consumers to generate interfaring request. We call the other consumers *Noise Consumers* (NC).

TABLE II shows the placement of consumers and producers in the four different configurations. MC, NC, and P stand for Main Consumer, Noise Consumer, and Producer, respectively. A consumer and a producer tied with "-" means the consumer requests contents served by the producer. The numbers attached to the prefixes MC, HC, and P expresses the node number in Fig. 8. Hash-Collectors are placed at all the nodes except the ones written in Table II. The distance threshold is used to determine *near*-ness among hash values. *MC-P hops* in the table expresses the average # hops between an MCs and P in the topology.

TABLE II
LOCATIONS OF CONSUMERS AND PRODUCERS

Configuration	locations
A	MC21-P30
B	MC17-P35
C	MC17,31-P35
D	MC17-P35, NC12-P1, NC31-P36

D. Results

The experiment is conducted under combinations of alternatives of (a)/(b) and (1)/(2) mentioned in section IV-B and configurations A - D mentioned in section IV-C. Table III shows the average number of hops traveled by data packets in each experiment.

TABLE III
AVERAGE NUMBER OF HOPS OF DATA PACKETS

configuration	distance threshold (%)	avg. # hops of data packets			
		(a)(1)	(b)(1)	(a)(2)	(b)(2)
A MC-P hops: 4.0	50	3.529	3.934	3.393	3.335
	67	3.999	3.920	3.025	3.288
	80	3.985	3.888	3.354	3.607
	100	3.817	3.593	3.557	3.605
B MC-P hops: 6.0	50	4.864	5.886	4.942	4.830
	67	5.804	5.156	4.521	4.802
	80	5.954	5.962	5.701	4.665
	100	5.671	5.556	5.346	5.150
C MC-P hops: 5.998	50	5.148	4.596	5.911	3.881
	67	5.909	5.962	4.527	5.156
	80	5.854	5.284	5.409	5.801
	100	5.404	5.708	4.617	5.915
D MC-P hops: 6.0	50	5.913	5.950	4.963	5.925
	67	5.953	5.846	5.504	5.892
	80	5.949	5.948	5.161	5.256
	100	5.931	5.929	5.640	5.870

E. Discussion

Table III shows that Hash-Collecting System can shorten the number of hops from the ones without Hash-Collecting System which are expressed with "MC-P hops" in the table. What is common in all the results is that there is no correlation between the distance threshold and the number of hops derived.

In configurations A and B where only one pair of a consumer and a producer exists, the cache hits are caused by contents which are requested from Hash-Collectors. In these configurations, the numbers of hops are fewer in condition (2), than those in condition (1). It means that the probability of cache hit is higher when Hash-Collectors goes in search to farther nodes.

Comparing the results of configurations B and C, there is not much difference in the number of hops between the two results. Thus, the number of Main Consumers does not affect the behavior of Hash-Collecting System.

Comparing the results of configurations B and D, the effect of reduction of hops gets rusty in configuration D because Noise Consumers replace the caches collected by Hash-Collectors. However, the usefulness of Hash-Collectors are still remained.

V. CONCLUSIONS

In this paper, we propose Hash-Collecting System and evaluate its performance. The results show the proposal method contributes the reduction of the number of hops to be traveled by data packets. While Hash-Collecting System works in various arrangements of nodes, its performance degrade when many consumers are present and compete for cache space.

Placement of Hash-Collectors with proper Collector IDs at right places can further shorten the hops for content retrieval and is a topic of future research. Also, the cache policy, which decides to cache or evict the content whose hash value is far away from the Collector ID, is needed, because hash-interest packets can bring the contents whose hash value is not "near" to Collector ID from a producer.

REFERENCES

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. Briggs, and R. Braynard, "Networking named content," in *Proceedings of the 5th ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT 2009)*, 2009.

[2] "Named data networking project website," Last accessed on Nov. 2, 2019. [Online]. Available: https://named-data.net

[3] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A distributed anonymous information storage and retrieval system," *Lecture Notes in Computer Science*, vol. 2009, pp. 46+, 2001. [Online]. Available: http://citeseer.ist.psu.edu/clarke00freenet.html

[4] J. Cao, D. Pei, X. Zhang, B. Zhang, and Y. Zhao, "Fetching popular data from the nearest replica in ndn," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, 2016, pp. 1–9.

[5] A. Suwannasa, M. Broadbent, and A. Mauthe, "Vicinity-based replica finding in named data networking," in *2020 International Conference on Information Networking (ICOIN)*, 01 2020, pp. 146–151.

[6] S. Mastorakis, A. Afanasyev, I. Moiseenko, and L. Zhang., "ndnSIM 2.0: A new version of the NDN simulator for NS-3," Named Data Networking Project, Technical Report NDN-0028, January 2016, [Online]. Available: http://named-data.net/publications/techreports/ndn-0028-1-ndnsim-v2/.

[7] I. Murase, "Trends on network testbeds in the world," *Journal of the National Institute of Information and Communications Technology*, vol. 52, no. 3/4, pp. 21–30, September/December 2005.