Performance Evaluation of Centralized In-Network Caching and Content Visibility in Information Centric Network over SDN/OpenFlow

Chanthan Hel
Department of Electrical Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
hel.chanthan65@gmail.com

Chaiyachet Saivichit
Department of Electrical Engineering
Faculty of Engineering, Chulalongkorn University
Bangkok, Thailand
chaiyachet.s@chula.ac.th

***Abstract*—Performance is the main issue that we must consider as the key point in order to design Information Centric Network architecture (ICN). Cooperative in-network caching and ability of network's nodes to see the contents in network will improve performance of ICN. In centralized network, the controller plays very important role in order to enable ICN nodes to do cooperative caching. Also, it has ability to see contents partially or globally according to our configuration. This paper aims to observe and compare the performance of ICN when we apply different levels of cooperative in-network caching and content visibility based on centralized control by using SDN/OpenFlow concept. We will evaluate performance of ICN by using three mechanisms; firstly, non-cooperative in-network caching with individual visibility; secondly, path cooperative in-network caching with path content visibility; and finally, global cooperative in-network caching with global content visibility. Our emulation result shows that global cooperative in-network caching with global content visibility mechanism gives better performance for ICN in terms of average number of hops to reach the content and number of requests hit server.**

Keywords—Centralized in-network caching; content visibility; performance evaluation; OpenFlow; Software Defined Networking (SDN)

I. INTRODUCTION

Information Centric Network (ICN) is new network architecture for internet and is currently under studying. Many researchers and research organizations have proposed several ICN's architectures and concepts [1]. Moreover, networking technology is now moving to Software Defined Networking (SDN) concept, that network is separated into two planes, data plane and control plane. OpenFlow is one of SDN's protocol. OpenFlow concept allows us to implement the testbed to perform any experimentations easily based on centralized control [2]. Among ICN research groups; there is a team in OFELIA project that proposed ICN architecture under support of SDN/OpenFlow [3], [4], [5]. We adopt the concept of ICN architecture of this work for doing our cooperative in-network caching and content visibility mechanism study. However, performance is the key issue that we need to consider in order to design ICN architecture. There is a consensus among the researchers that in-network caching of ICN has influences on performance of the whole network. One of the most interesting topics associated with in-network caching is cooperative/collaborative in- network caching. Cooperative in-network caching is a mechanism that ICN nodes inside the network do collaboration with each other to cache contents in order to improve the network's performance. There are several levels of cache cooperation such as partial, path and global cooperation. Furthermore; content visibility, the ability of ICN nodes to be aware of locations of contents in the network, also affects the whole network performance [6]. When ICN node knows more about the locations of contents, the performance of network can be enhanced because ICN nodes can forward requests to reach contents' locations more effectively. In centralized network, cooperative in-network caching and content visibility mechanism will be implemented in network controller. The controller will decide where to cache data contents and have ability to look for data corresponding to request packet. The controller can be configured to have three varieties of mechanisms. Firstly, the controller cannot see all contents in the network and ICN node will do in-network caching by their own policy but the controller stores all the routes for forwarding contents in network, we can call this mechanism as non-cooperative in-network caching with individual content visibility. Secondly, the controller has ability to command only ICN nodes along the path from requester to server to do cooperative in-network caching and it can search location of content along the same path only, we call this mechanism as path cooperative in-network caching with path content visibility. Lastly, the controller can do in-network caching and search for locations of contents globally, we call it as global cooperative in-network caching with global content visibility. In our work, we want to evaluate the network performance among those three mechanisms based on centralized-control using OpenFlow.

There are several kinds of cooperative in-network caching such as partial, path, neighboring and global cooperation [6]. Most of the works are concerned about distributed and decentralized network. Routers in the network normally make decision by themselves whether to cache the contents passing through them according to information they get from other nodes [7], [8]. Many works consider only partial and individual visibility of contents [7], [8]. It is difficult to find the works that take global content visibility into account in their studies.

Moreover, works that compare performance of ICN by applying various cooperative in-network caching and content visibility mechanisms are also uncommon. Reference [9] compares performance of ICN based on three cache placement policies: Least Recently Used (LRU), single-path and network-wide caching policies. LRU is a manner of non-cooperative caching whereas single-path is a caching policy that ICN nodes along the path from requester to server will do collaboration together to cache contents. Moreover, for network-wide, all ICN nodes inside network will do cooperation in order to cache contents. They also use Mixed Integer Programing to calculate where to cache contents in order to minimize delivery cost. But, this work is studied in a decentralized manner and every ICN node can only see contents in its own cache. According to [6], it is not easy to apply global content visibility in decentralized network. So, centralized-control network is the best choice. All ICN nodes just update their contents' information to control plane and mechanisms of global content searching can be implemented in the controller. However, [10] applies quite similar concept that we want to do. It aims to study on advantages and disadvantages of web cache collaboration in centralized-control network. The performance is evaluated at various levels of web cache collaboration such as no caching, independent caching and cooperative caching. The result of this work shows that cooperative caching with centralized control gives better performance compared to independent and no caching of web page in backbone network. Unlike our work that performed study on ICN, the network studied in [10] is IP-based network.

In our work we plan to evaluate the performance of ICN when we apply different levels of cooperative in-network caching and content visibility using centralized control of SDN/OpenFlow. We will study this in only one autonomous system and message content shall be sent as one entity, not to be fragmented.

In section II, we will demonstrate about ICN architecture over SDN/OpenFlow concept that we will evaluate performance in this paper. Detail of operation for various centralized in-network caching and content visibility mechanisms is in section III. Performance evaluation and result discussion are in section IV. The last part is conclusion of the work.

II. Information Centric Network over SDN/OpenFlow

This architecture is based on the concept of Software Defined Networking (SDN). There are two planes in network, control and data plane. Those two planes will communicate with each other by using OpenFlow protocol. The main component of data plane is ICN nodes and the main component of control plane is network controller or controller.

A. Controller

It is an ICN's component that controls all ICN nodes in data plane. It stores all routes for forwarding contents in the network. It collects information of contents from every ICN nodes. It knows the topology of network. Based on this capability, the controller can use this information in order to determine actions on data and interest/request packets such as which route is the most appropriate for packets to reach the destination or which ICN node should cache which content. Note that data packet is a packet that contains data of content and interest/request packet is a packet generated by requesters to fetch data packet. In this work, the controller will calculate the shortest path from requester to ICN node storing data corresponds to request packet and packets will be forwarded along that path.

B. ICN Node:

It can be router or switch. It also has ability to store contents because it has been equipped with cache inside. Its role is to forward packets, to reply data content to requesters if it has data corresponds to the interest. It also applies actions on packets based on action set receiving from the controller. Figure 1 shows about architecture of ICN over SDN in one autonomous system.

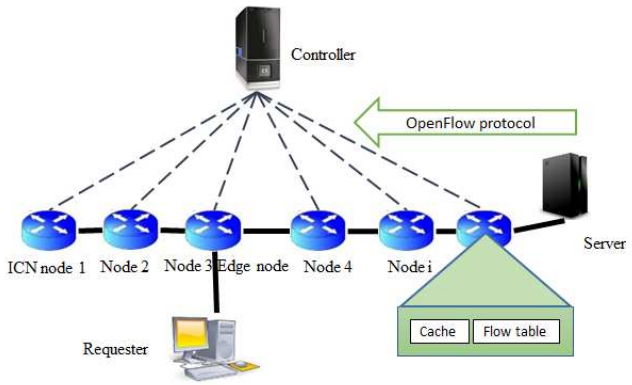

Fig. 1. ICN over SDN/OpenFlow architecture for one autonomous system

III. Operation of Various Centralized In-network Caching and Contetnt Visibility Mechanisms

This section will explain the operations for three centralized in-network caching and content visibilities mechanisms that we consider in the performance evaluation of ICN. The difference between these mechanisms is the ability of the controller to see contents in network, to command ICN nodes to perform cooperation to cache. We know that one of advantages of centralized network is that the controller can obtain all information from the network regarding to contents' information, contents' location, and it would be easy to implement any algorithm in the controller. However, ICN is a network that focuses on contents rather than addresses of contents, so global visibility of contents or global cooperative in-network caching by using centralized controller may be a challenge. It needs very fast processors of the controller. This work shall evaluate the performance of ICN based on three mechanisms: Non-cooperative in-network caching with individual content visibility; path cooperative in-network caching with path content visibility; and global cooperative in-network caching with global content visibility.

A. Non-Cooperative In-network Caching with Individual Content Visibility Mechanism

This mechanism is very simple. The controller in this network just only stores the routes to forward contents in the

network, shortest path for our work. When interest packet arrives at ICN nodes, nodes will check data content for this interest packet. If data content is in their caches, they will reply data to requester and if there is no data in caches they will ask the controller which port this request should be sent to get the shortest path to server. That outcome will be stored in ICN nodes. ICN nodes will know where to forward the request to without asking the controller again later. On the other hand; every ICN node along the path from node that replies data to content requester will copy every data packet. If their caches are full, one content will be removed randomly from cache and replaced by new content. It is the manner of random replacement policy and copy everywhere placement policy.

B. *Path Cooperative In-network Caching with Path Content Visibility Mechanism*

This is a mechanism that the controller chooses only the ICN nodes along the shortest path from requester to server to cooperate with one another to cache contents/data. Also, the controller can only search or see location of data corresponds to request packet along the same path. In Figure 1, when an interest packet first reaches the ICN node 3 that is connected to requester (or we may call it "edge node"), that interest packet's header is sent to the controller. The controller, then, determines the shortest path from requester to server, for example, the route from node 3 to the server. After that, the controller checks where the data of this interest packet is. It checks data in the list obtained from edge node, then checks in lists obtained from other ICN nodes along the path in the direction to server until it can find the data packet. In case data packet is previously stored in the edge node's cache, the controller will instruct edge node to reply with the data back to the requester. If the required data packet is not in edge node, but it is located in node *i*, the controller will command set of actions to edge node to modify header's field of the interest packet by adding an identifier to define that data of this interest is in node *i*. So, when ICN nodes along the path between edge node and node *i* receive interest packet, they will check their flow table in order to forward packet. If they cannot find any actions to do on the packet, then interest packet's header will be sent to the controller. The controller will then issue a command to nodes to forward this packet and store the route information in the nodes' flow table. In another case where the controller cannot see the data packet in network, it will send instead the route to forward the interest packet to the server.

When an interest packet reaches the node containing data, its header is sent to the controller. The controller decides where to cache that data packet along the path. It depends on the algorithm that we implement for in-network caching in the controller. For our work, we use popularity based caching policy as described in [11]. We assume that we know the request pattern, so the most popular contents will be cached at the edge node and other nodes where located one hop or two hops away from edge node. The less popular contents will be cached farther from requester. On the other hand, if the data is from server and arrives at the ICN node connected to the server, that node will send packet's header to the controller. The controller will decide where to cache that data in the same manner as we mentioned above. Note that for this mechanism, the controller can only decide where to cache the data and search for data in the ICN nodes along the path from the server to the requester as determined by the controller at the edge node when the interest packet first arrives.

C. *Global Cooperative In-network Caching with Global Content Visibility Mechanism*

In this mechanism, the controller has an ability to see locations of all contents in the network and to decide where to cache data packet in any nodes inside the network. That is why we call global cooperative in-network caching with global content visibility mechanism. When an interest packet reaches edge node, its header is sent to the controller. The controller then searches globally for the locations of the data packet according to the interest packet. First of all, the controller checks for that data in the content list obtained from edge node, and then it checks in the lists obtained from every node that is one hop, two hops, and so on, away from edge node. The searching mechanism is finished when the controller finds location of the data content. In case that data is not in the network, the controller will choose the shortest path to send that interest packet to the server. In another case where the data content is in the network but not at the edge node, the controller will calculate the shortest path from the requester to the node that stores the data. The controller will order the edge node to put an identifier in the header field of the interest packet before sending to next node. When the packet arrives at the next node, for example node 2 in Figure 1, node 2 then checks its flow table whether anything matches the request of that packet. If it cannot see any route to forward that packet, it will send the header content to the controller. Then, it applies action directed from the controller on the interest packet and sends the packet to next hop. The operation is repeated until the interest reaches the node that stores the required data.

When an interest packet reaches node containing data packet, the interest packet's header is sent to the controller. At this point, the controller makes a decision on where to cache the data content in the network. In this mechanism, the data content is not only cached at ICN nodes along the path from the requester to the node that stores content, but it can also be cached at any other nodes in the network. When the decision of content caching is done, the controller instructs ICN node to reply data and forward to the requester along the path that interest packet travels. An identifier is put in the data packet's header in order to indicate which node to cache the content. If the controller decides to cache the data packet in several nodes in network, the controller will instruct ICN node that stores the data to copy several packets then send to various directions. The controller will help ICN nodes to forward data packet to deliver to the requester and the nodes that need to cache such data packet. In our work, we have defined that the most popular contents will normally be cached near the requester.

IV. PERFORMANCE EVALUATION

A. *Performance Metrics*

To evaluate performance of ICN over three mechanisms of centralized in-network caching and content visibility that we just mentioned in section III, we decide to choose two metrics:

1) Number of request packets hit server: This is a metric that aims to count total numbers of interest packets that will get data from server. When the number of interest packets hit server is low, so it means that most of requests are served by in-network nodes.

2) Average number of hops to reach the content: It refers to average number of hops that one interest/request packet needs to pass through in the network in order to get the data packet. The smaller it is the better performance is achieved.

B. Experimental Setup

In this work, we perform experiment by using cascade network topology that consists of six ICN nodes, one server, one controller and two requesters as shown in Figure 2. We used random replacement policy to replace content when the cache of ICN node is full because cache replace policy do not affect significantly the network performance [12], [13]. We used popularity based caching policy. On the other hand, requesters request the contents based on Zipf popularity distribution [11] with exponent $\alpha = 2$. All parameters for experimentation are listed in Table I.

TABLE I. PARAMETERS USED FOR EXPERIMENTATION

Parameters	Values
# of total contents	1000 contents
# of content classes	10 classes
# of contents per class	100
Request rate for each requester	λ = 5 requests/sec
Content size	1Kbyte
Total requests for each requester	5000
Zipf exponent	α = 2

Mininet [14] is selected as network emulator for our experiment. ICN nodes, requesters and server in data plane are created in Mininet VM image 2.1.0 running in a virtual box in Core-2-Duo-2.8 GHz Dell machine. Then, we run POX controller [15] in a Dell machine.

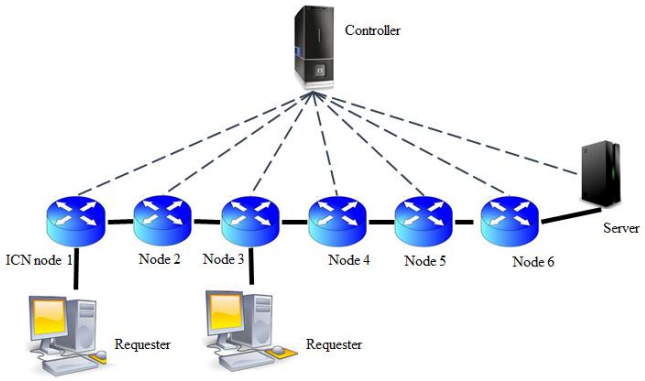

Fig. 2. Cascade topology with 6 ICN nodes, 2 requesters

All ICN nodes and server are Open vswitch [16]. Requesters run in virtual hosts created by Mininet. We use Scapy [17] to generate interest packets. In our experiment, ICN nodes do not store real contents. They only apply OpenFlow action message from the controller on the packet. All the list of contents will be stored in the controller. On the other hand, we use IP packets as interest and data packets for ICN in our experiment. We separate these kinds of packets by using MAC address as a tag. The request packet is a packet that contains the MAC address of the requester as its source MAC address. The data packet is a packet that contains requester's MAC address as its destination MAC address. Furthermore, we use UPD source port to identify the name of content and UDP destination port as the identifier for packet to be cached at any node as we mentioned in section III. At the beginning, all caches of ICN nodes are empty. We will observe the performance of ICN with different cache sizes also.

C. Emulation Results and Discussion

After finishing the emulation, we obtained the average number of hops to fetch the content for every centralized in-network caching and content visibility mechanism in terms of cache size as shown in Figure 3.

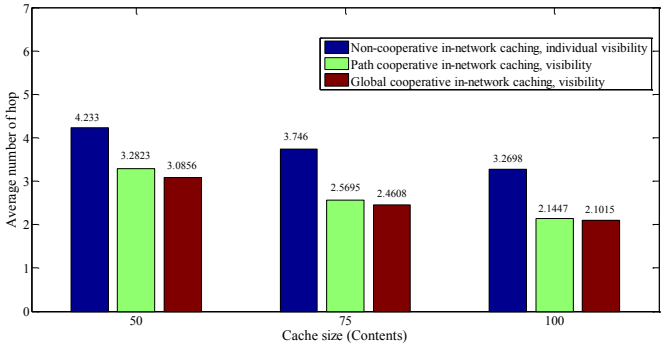

Fig. 3. Average number of hops to reach the content

We observe that for every cache size of ICN node, global cooperative in-network caching with global content visibility mechanism always outperforms other mechanisms. The average number of hops to reach the content decreases when size of ICN node's cache increases. Because when cache size is large, it can store more contents in the network. Moreover; we see that when the cache can store contents up to 100, average number of hops to reach the content of path and global cooperation mechanism is almost the same. This happens because we use the popularity based caching policy to cache content. The most popular content will be cached at the ICN nodes that are nearest to requesters. Request pattern follows Zipf popularity distribution with $\alpha = 2$, it means that around 65% of the requests are of class number one of contents that consists of 100 contents. So, most of the requests will meet data contents at one hop and two hops away from the requesters. If request pattern changes, we believe that the result will change either.

If we look at the numbers of requests that hit server in Figure 4, we notice that 57% of request packets are served by server for non-cooperative in-network caching mechanism while there are only 37% and 27% for path and global cooperative in-network caching mechanism respectively when cache size is equal to 50 contents. Again when cache size is large, number of request packets that reach server is almost the same for path and global cooperative in-network caching

around 14% of total requests. However, we observe that global cooperative in-network caching with global content visibility mechanism still outperforms other mechanisms. Because the controller knows where to cache contents in order to get better performance and it can search for contents' locations in the whole network. The controller can forward request packet to the nearest ICN node that stores data. The result above may be valid for only small and simple network topology as in our experimental setup.

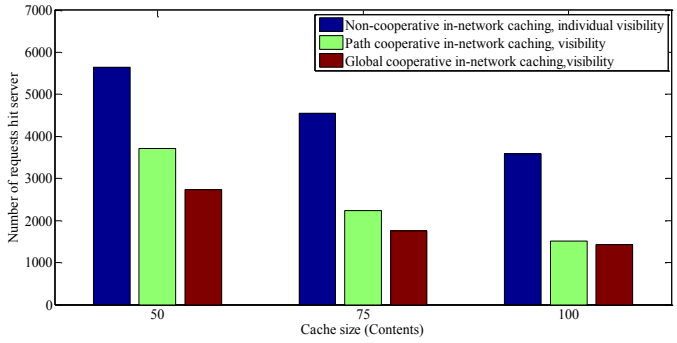

Fig. 4. Number of request packets hit server

However; although the global cooperative in-network caching with global content visibility mechanism gives better performance in terms of average number of hops to reach the content and number of requests hit server, but we think that algorithm for deciding where to cache content and for searching content globally is a very complicated task, especially when number of contents in network are large. On the other hand, to make result more applicable, we need to take more performance metrics in to account such as expected round trip time to retrieve content and communication messages between control and data plane. These metrics will give us more information about the delay of centralized system over SDN/OpenFlow and messages stress between control and data plane. However, network topology is also the main factor that we need to consider. Anyway, we believe that in order to compare and evaluate performance of the three mechanisms above more effectively, we should perform the experiment with the real OpenFlow testbed. Now, we are also implementing the real testbed to perform the experiment.

V. CONCLUSION

This paper aims to observe and compare performance of ICN when we apply several levels of cooperative in-network caching and content visibility based on centralized control by using SDN/OpenFlow concept. According to our investigation; we see that the controller that has ability to do in-network caching and to see content globally, we called global in-network caching with global content visibility mechanism, can improve network performance by reducing number of requests hitting server and average number of hops to reach the content. It gives better performance while comparing to non-cooperation and path cooperation mechanism. However, this result is applicable for only small and simple network topology as in our experimental setup. To make the result more accurate, we plan to run the experiment of these mechanisms again in the real OpenFlow testbed. We also plan to evaluate ICN's performance in various network topologies such as tree and arbitrary topologies. We also plan to do performance evaluation study based on the three mechanisms perform in our work in medium size of ICN and we will take more testbed performance metrics into account.

REFERENCES

[1] B. Ahlgren, C. Dannewitz, C. Imbrenda, D Kutscher, and B. Ohlman, "A survey of information-centric networking," Communications Magazine, IEEE , vol.50, pp. 26-36, July 2012.

[2] N. McKeown et al. "OpenFlow: enabling innovation in campus networks". White paper, March 2008.

[3] L. Veltri1, G. Morabito, S. Salsano, N. Blefari-Melazzi and A Detti, "Supporting information-centric functionality in software defined networks," in Communication IEEE International Conference, pp. 6645-6650, June 2012.

[4] N. Blefari-MELAZZI, A. DETTI, G. MAZZA, G. MORABITO, S. SALSANO and L. VELTRI, "An OpenFlow-based testbed for information centric networking," in Future Network & Mobile Summit, pp. 1-9, July 2012.

[5] S. Salsano, N. Blefari-Melazzi, A. Detti, G. Morabito, L. Veltri, "Information centric networking over SDN and OpenFlow: architectural aspects and experiments on the OFELIA testbed," Computer Networks, vol. 57, pp. 3207–3221, November 2013.

[6] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," Computer Networks, vol.57, pp.3128–3141, August 2013.

[7] Z. Ming, X. Mingwe, and D. Wang, "Aged-based cooperative caching in information centric networks," INFOCOM WKSHPS, pp. 268-273, 2012.

[8] L. Zhe and S. Gwendal, "Time-shifted TV in content centric networks: the case for cooperative in-network caching," Proc of IEEE Int. Conf. on Communications (ICC), 2011.

[9] Y. Kim and I. Yeom, "Performance analysis of in-network caching for content centric network," Computer Networks, vol. 57, pp. 2465–2482, April 2013.

[10] S. Paul and Z. Fei, "Distributed caching with centralized control," Computer Communication, vol.24, pp. 256-268, 2001.

[11] K. Suksomboon et al. "PopCache: cache more or less based on content popularity for information centric networking," Local Computer Networks , pp. 236-243, 2013.

[12] H. Jeon, B. Lee and H. Song, "On-path caching in information centric networking," ICACT, pp. 264-267, 2013.

[13] D. Rossi and G. Rossini, "Caching performance of content centric networks under multi-path routing (and more)," Technical report, Telecom ParisTech, 2011. Available from: http://www.enst.fr/~drossi/paper/rossi11ccn-techrep1.pdf

[14] Mininet. http://mininet.org/

[15] POX. http://www.noxrepo.org/pox/versionsdownloads/

[16] Open vswitch : http://openvswitch.org/

[17] Scapy. http://www.secdev.org/projects/scapy/