Efficient Forwarding and Popularity Based Caching for Content Centric Network

Kyi Thar, Thant Zin Oo, Chuan Pham, Saeed Ullah, Doo Ho Lee, Choong Seon Hong

Department of Computer Engineering, Kyung Hee University,
446-701, Republic of Korea
Email:{kyithar, tzoo, pchuan, saeed, dooholee, cshong}@khu.ac.kr

***Abstract*—In Content Centric Network, caching and forwarding schemes can affect the performance of the whole network. The original caching and forwarding schemes are simple, but these schemes have some drawbacks. Firstly, the original caching scheme stores the same content on several neighboring routers along the request path. This redundant caching does not use the limited storage space available to the routers efficiently. Secondly, the original forwarding scheme in which the data requests are flooded to neighboring routers, also degrades the performance of the network. In this paper, we aim to solve the issues of using cache space of each router efficiently and forwarding the data requests effectively. In this proposal, we divided an Autonomous System (AS) in several groups of routers to cache the popular contents. Routers in a group cooperatively store the data and forward the Interest in order to increase the network performance. To improve the cache hit, the group of routers only store the popular contents and reduce the duplicate content without effecting the redundancy. We used Consistent Hashing to reduce overlapping contents and forward the request efficiently. The content popularity prediction algorithm assists the routers to store the popular contents that pass through them. Finally, we evaluated the performance of our proposed scheme by using a chunk level simulator.**

Keywords*—*Content Centric Network, Popularity based caching, Cooperative Forwarding.

I. Introduction

According to the Cisco survey, most of the Internet traffic is occurred by watching Internet videos. To reduce the traffic inside the Autonomous System (AS), Jacobson et al. have proposed the Content Centric Networks (CCN) or Named Data Networking (NDN) [1]. CCN changes the current Internet concept, which is location based, to information specific. The main feature of a CCN router is; to store Data (content object) temporarily and give the copied of stored Data to the users when they request. That is why, by deploying CCN routers, the traffic inside the network and also the fetching delay will be reduced.

There are two types of packets in CCN, Interest (packet) requests the Data (packet) and Data (it is also known as Content object) is the reply packet for the requested Interest. Whenever a CCN router receives the interest packet, it checks three things: Content Store (CS), Pending Interest Table (PIT) and Forwarding Interest Based (FIB). CS is the cache memory of router that stores copies of the passing data temporarily to provide it to the future requests. Data is stored in CS in the form of a sequence of segmented Data, which is denoted as chunks. One interest can request only one chunk. If the requested data item is found in the CS it is immediately provided to the users. If data item is not founded in the CS then the router checks PIT, which is a table that keeps the list of interests that are already requested and waiting for receiving the corresponding data item. If no entry is found in the PIT for the received interest packet, the router searches the FIB, which stored the lists of the incoming and outgoing Interface, in order to find the best path to the destination. If no entry for the desired destination is found then the interest packet is flooded in the network through all the interfaces.

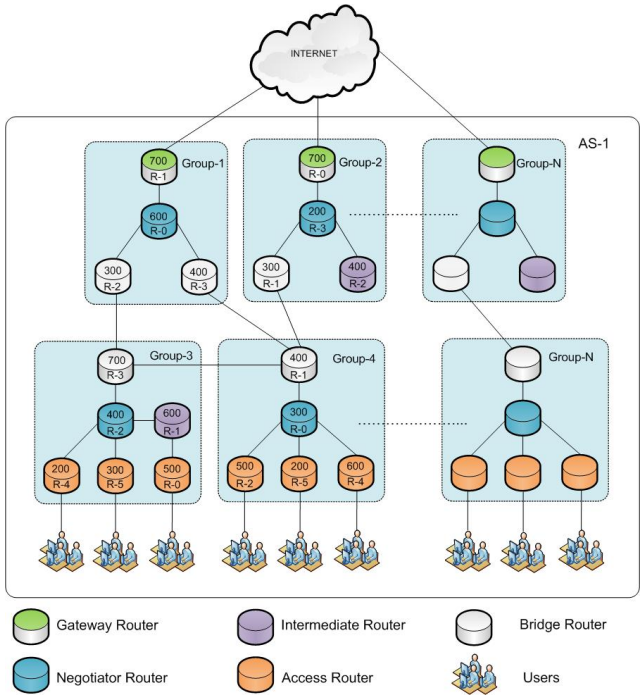

Fig. 1. System model

In a CCN router, the following strategies are necessary; cache decision, cache replacement and forwarding strategy. Cache Decision strategy defines a policy to make a decision whether to store Data (Content Object) on current router. The Cache Replacement strategy defines which content should delete to create the free space, when the cache space of the router is full. The Forwarding strategy defines how to forward the Interest to neighboring routers if the requested content is not located on the current router.

As the cache space of each router is limited, to achieve

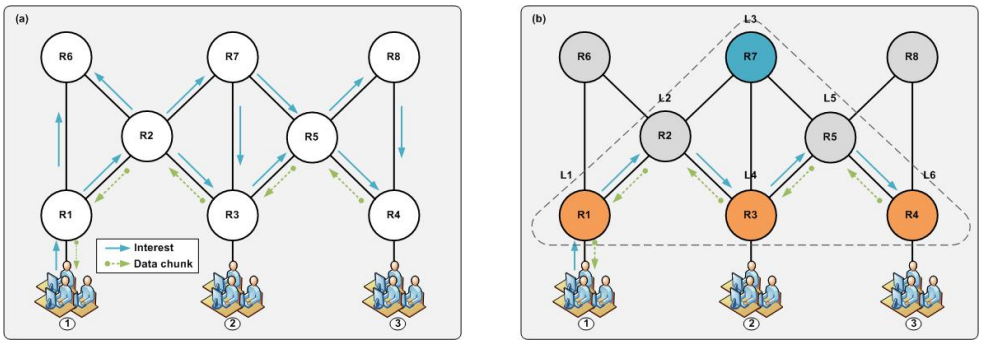

Fig. 2. The comparison of original Forwarding scheme and proposed Forwarding scheme. (a)For original Forwarding, user (from group 1) requests the content by sending Interest to R_1 and R_1 does not has requested content. So, R_1 floods the Interest. R_2,R_3,R_5,R_6,R_7,R_8 also flood the Interest. When R_4 (It has requested content.) receives the Interest and R_4 replies the content by PIT table. The copy of reply content is saved on the routers R_2,R_3,R_5,R_6,R_7,R_8. (b) For proposed Forwarding,R_1,R_2,R_3,R_4,R_5,R_7 are within one group. User (from group 1) request a content to R_1 and it finds the custodian of the content by using hash function. R_4 is the custodian of the content.R_1 forwards the request to R_2 and R_2 chooses the best neighbor router R_3. R_3 chooses R_5 and R_4 receives the request. The copy of reply content will not be saved on the reply path.

efficient caching, neighbor routers should not store the same content. At the same time, routers should forward the Interest efficiently to reduce the burden of matching incoming Interest with CS, PIT and FIB on each router. (The computational overhead for matching the interest in three tables can be significant.). We observe that a cooperative mechanism is needed to reduce the duplicate contents without effecting the redundancy and to forward the Interest efficiently.

This proposal consists of three parts. Firstly, only a single copy of content will be cached inside one group of routers. In this way, we can save cache space without effecting the redundancy of the contents. Secondly, contents are stored based on their popularity. We used popularity prediction algorithm that predicts the future popularity of the contents depending on the old popularity value and the current one. Thirdly, to solve the scalable and load balancing problem of the network, we employ consistent hashing to forward the Interest and cache the Data.

II. Related work

There are two types of caching, on path caching and off path caching. Data (content objects) are stored on each content router that are on the request path and it is also known as on-path caching [6]. If the content objects are cached on corresponding router and it is known as off-path caching.

Leave Copy Everywhere (LCE) is the original caching strategy of CCN which caches the data on each router that is located on the Interest request path. LCE is good for redundancy of the data but the usage of the cache space is inefficient. Leave Copy Down (LCD), which is proposed in [9] for web caching, reduces the duplicate data on the request path. However, the reduced redundancy degrades the cache hit. Progressive Caching [12] extends the LCD to cache popular chunks and solve the problem of one timer cache. In [11], used probabilistic caching scheme where the closer the router is to the user, the higher the probability for caching the content

In default CCN, a router floods the Interest to the neighbor routers to find the Data. Flooding the Interest degrade the performance of the network. In [13], [14], a router ranked the interfaces and choose the best path to retrieve Content Objects. If the requested Data is not on the request path, routers need to choose new interfaces to forward that Interest again. In [8], the authors try to solve the caching problem of the duplicate chunks around the one hop neighbors by exchanging the cache summary periodically and eliminating the duplicate contents. In [10],authors introduced the Availability Info Base (AIB) to know which router stores which Data. Modulo hashing is widely used in cooperative caching and forwarding [5] [6] [7].

III. Efficient Forwarding and Content Popularity Prediction Based Caching for CCN

This section consists of four parts. Firstly, we discuss about how to construct the hash ring map by using consistent hashing. The virtual routers concept is used to balance the load of the routers inside one group. The second part explains the popularity prediction algorithm that support Cache Decision Strategy to store the popular chunks. The third part describes Cache Decision algorithm that supports the group of routers to store the popular contents. The fourth part defines the Co-operative Forwarding Algorithm, which forwards the request to designated routers without flooding.

A. System model

Fig.1 shows the system model of the proposed scheme. CCN routers are formed into a group by system administrator or any clustering algorithm. One AS consists of several groups and different groups consist of different number of routers. The cache size of the routers are heterogeneous. Each physical router possesses the keys or virtual routers depending on its physical cache capacity as shown in Fig.3. A router will possess more keys or more virtual nodes than others when the cache size of the router is bigger than others. For example, R_1 possesses the five keys(2,6,7,10,12,13) and this is also mean that the router processes five virtual routers. Router hash the Interest name with hash function and stores the Data that hash value is 2,6,7,10,12,13. So, R_1 is the custodian router of these Data. Routers inside one group know each other by sharing hash ring map as shown in Fig 4. So every router inside one group know the custodian of the requested content. Each group consists of three types of routers, Negotiator Router (NR), Bridge Router (BR) and Gateway Router (GR). BR connects with the other cluster and GR connected with the other AS(s).

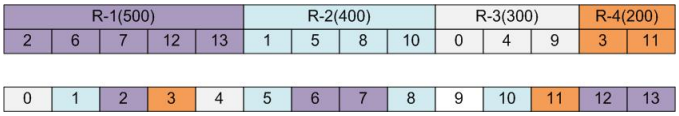

Fig. 3. Cache size aware key distribution for one group of router. One physical router can possess several random keys or virtual routers depending on its physical cache size of router.

B. Construction the hash ring with virtual nodes

The reason for constructing the hash ring is to eliminate the duplicate chunks in an on-line manner and to maintain the balance of the load. At the same time, hash ring can be used for efficient Interest forwarding and also for Data retrieving.

Modulo hashing is simple and widely used in forwarding and making cache decisions. In modulo hashing (key modulo n), the number of physical routers (n) are used as divisor and the incoming Interest (key) as dividend. It is light weight hashing scheme but it has consistency and load balancing issues. Even if one router is added or removed from the group, the location of the all previously cached data are changed and these objects become useless. If the routers hash the sequence number of chunks by modulo hashing, some routers will be dumped with more Data and several routers have small loads. Further more, the usage of cache space is inefficient and load distribution is unbalanced. To solve this problem, we used consistent hashing instead of modulo hashing.

The consistent hashing uses keys range or number of virtual routers as a divisor that is independent from the number of physical routers. In here, keys range is considered as a ring with values from (0 to n virtual routers). In original consistent hashing [3], routers possess the random points on the hash ring. That kind of mechanism can not control the load balance of the network. Some router may handle large keys space and some may small keys space, whatever the physical router cache size is large or small. There is also another issue, when the router joins or leave from the group, the key space handled by the router will dump onto the remaining node that is located on the first position in close-wise direction. This is not fair, when the small cache size router handle the large key space. To solve this problems mentioned above, we construct the virtual routers like [4].

In this paper, one group of routers possess the n number of keys range (0,1, 2...n) and each key represents one virtual router. One physical router can possess several random keys or virtual routers depending on the capacity of physical cache size. The procedures for constructing the consistent hash ring are as follow. At the initial stage, groups or clusters are manually formed by the administrator or any clustering algorithm. After this stage, the node which has highest betweenness centrality value becomes NR. Routers inside one group construct the Neighbor Router Group Table (NRGT) to knows how the routers are connected and by which interfaces. NRGT consists of three fields: current router label, destination router label and interfaces. NR collects the group information and construct the hash ring map. In here, routers possess the keys range that are the points on ring. By assigning several random point to each physical router within one group, we form a hash ring map. For example, shown in Fig 3, router 1 possesses the five keys and that keys will be distributed randomly (2,6,7,10,12,13), router 2 possesses four keys (1,5,8,10), router 3 possesses three keys

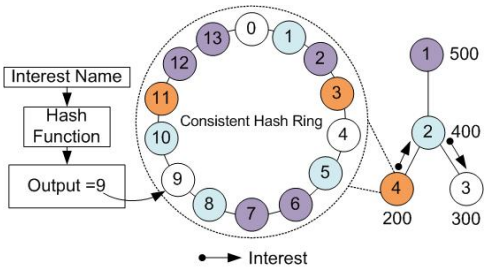

Fig. 4. Consistent hash ring with virtual routers: by using this, each router inside one group can know the custodian of the incoming Interest. In this figure, router 4 finds the custodian of an Interest by using hash function.

(0,4,9) and router 4 possesses two keys.

If a router failure occurs, then the points for that router will be handled by the other existing routers in that group. Each router handles random distributed keys. So, the incoming Interest and caching Data for the failed router would have mapped to the next highest point router. However, the key values mapping to the other router will remain the same. In this way, when the R_1 fails, the keys possessed by that router will not dump onto only one router. The process is similar, when a new router is added to the group. It means, several random point are added to the ring. After that, these points will no longer be possessed the previous router. The Data previously stored on the previous router with those keys will be deleted by LRU policy. After constructing hash ring map, NR distribute that map to all members inside one group. The result is that all members know, the custodian of the requested Data and each router stores all the chunks depending on it keys.

C. Popularity prediction

The popularity prediction algorithm (algorithm 1) assists the Cache Decision Strategy to store the popular contents. The prediction algorithm for CCN should simple with low complexity. Most of the paper used simple content access counts and if the access counts is greater than the threshold , these contents are stored on the router. Finding the threshold is also an important issues to store the popular contents and the best threshold is computationally intensive. The threshold to cache popular content is a static one in [20]. The static threshold can not cached the popular contents, when the number of request increase in time t. So, in this paper, we used a dynamic threshold which changes depending on the number of requests.

Algorithm 1 Popularity Prediction

1: Input:schedule $t = \gamma$
2: Every time t, update the Popularity of contents with equation 1
3: Every time t, update the threshold with equation 2

Firstly, there are two parts in equation 1, which predicts the popularity of each content depending on the history and current request frequencies. This formula is taken from TCP RTT estimation [21]. β is the weight parameter, where the value is between 0 and 1. The router periodically calculates the popularity of the contents that passing through the router. In here, network administrator needs to tune the schedule to update the popularity of the contents and also the weight β.

The $\beta * \frac{F_i}{\sum_{i=0}^{j} F_j}$ considers the current access frequencies of the content. F_i is the access frequency of the content i and $\sum_{i=0}^{j} F_j$ is the total access frequency of the all content at time t.

Secondly, we will discuss about finding the threshold to cache the popular contents. In equation 2, θ_1 is the threshold at time t, $\sum_{i=0}^{j} LP_j(t+1)$ is the total predicted popularity values. $\sum_{i=0}^{j} C_j$ is the total number of contents at time t. In this way, the threshold is dynamically changed, depends on the number of request.

$$LP_i(t+1) = \beta * \frac{F_i(t)}{\sum_{i=0}^{j} F_j(t)} + (1-\beta) * LP_i(t-1), 0 < \beta < 1. \quad (1)$$

$$\theta_1(t) = \frac{\sum_{i=0}^{j} LP_j(t+1)}{\sum_{i=0}^{j} C_j(t)}. \quad (2)$$

D. Cooperative Forwarding

For cooperative forwarding and caching(section III-E), we introduce a new field in the PIT. This is done to known the custodian of the return Data, and this field is called Cache Bit (CB). There are two possible cases for caching and forwarding process. For the first case, when the CB bit is one the current router will cache the Data. For the second case, when the CB bit is $zero$ the current router will not cache the Data. To know the custodian of the incoming Interest, the router hashes the Interest name and checks the hash value, when it receives an Interest.

For the first case, the current router is the custodian of the requested Interest. So, the current router searches the requested content in the CS. If the requested content is not in the CS, the router needs to forward this Interest to the neighbor routers. Before forwarding the Interest, the router checks the PIT and FIB tables to know whether the Interest is already forwarded to the neighbor routers or not. If the requested Interest is not in the PIT and FIB lists, the router changes CB bit to one and then the update the PIT and FIB. After updating the PIT and FIB, the current router forwards the Interest to neighbor routers. If the Interest is already forwarded, router stops the forwarding process.

For the second case, the current router is not the custodian of the request Interest. In here, our proposal can reduce one step. The current router will not check the requested content in the CS and will directly forward it to the neighbor routers. Before forwarding the Interest, the router checks the PIT and FIB tables to know whether the Interest is already forwarded to the neighbor routers or not. If the requested Interest is not in the PIT and FIB lists, the router changes CB bit to $zero$ and then the update the PIT and FIB. Otherwise, the router stops forwarding Interest.

The cooperative forwarding is presented in algorithm 2, where I represents the Interest and I' represents the DF enable interest. R_k^i represents the router k in i^{th} group. R_C^i represents the current router in i group.

Algorithm 2 Cooperative Interest Forwarding

```
Input: I = Interest or I' = Updated I (Direct Forward)
if Incoming Interest is I' (Requested content is not in custodian router) then
    Current router checks I' in PIT list
    if The I' is already listed in PIT then
        Current router stop the forwarding process of I'
    else
        Add I' to PIT list and forward I' to neighbor router groups/AS via BR or GR
    end if
else
    if Incoming Interest is I (New request) then
        Find the Custodian router of I
        if Current router R_C^j is the Custodian router then
            Search I in it's Content Store (CS)
            if request object(O_i) is in the CS then
                Immediately reply O_i to the requested router
            else
                Current router checks I in PIT list
                if The I is already listed in PIT then
                    Current router stop the forwarding process of I
                else
                    update I to I'
                    Add I' to PIT list and set Cache bit to 1
                    Forward I' to neighbor groups/AS via BR or GR
                end if
            end if
        end if
    else
        Current router R_C^j is not custodian router
        Checks I in PIT list
        if The I is already listed in PIT then
            Current router stop the forwarding process of I'
        else
            Add I to PIT list and set Cache bit to 0
            Forward I to designate router R_k^j (custodian router)
        end if
    end if
end if
```

Solving the problem of Interest drop by intermediate router: The intermediate router will drop the Interest when the same Interest is already in PIT list. For example, in Fig.2(b), user sends a request(Interest) to router R_1 (whose label is 1). Router R_1 searches the custodian router of the requested Interest by using consistent hash ring. The custodian router of the Interest is R_4 (whose label is 6). The Interest is forwarded to the custodian router R_4 via R_2 (label 2),R_3 (label 4) and R_5 (label 5). In here, R_1,R_2,R_3 and R_5 will not be check the incoming Interest in their CS because these router are not custodian router of that Interest. So, these routers can reduce CS longest prefix matching time, which is significant, especially for an almost full CS.

The Interest is received by custodian router R_4 and the requested content is already deleted in its CS by LRU policy. So, this router forwards the Interest to BR router R_5. In this case, the BR router R_5 will drop the Interest because the

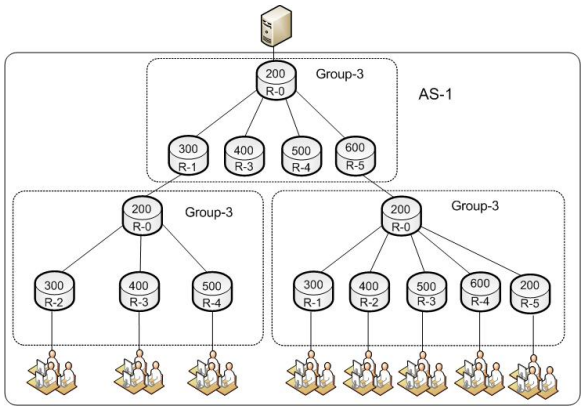

Fig. 5. Topology for simulation

Interest is already listed is in the PIT. In [5], the authors solved this problem by using piggy Interest. Any scheme which uses any hashing can be faced with this problem. In this paper, to prevent the blocking Interest by an intermediate routers, we add Direct Forward (DF) field in the Interest packet. If the DF value is 1, the intermediate routers will not be blocked and it will add that Interest to PIT and forwards it through corresponding face.

E. Cache Decision

We have already mentioned about Cache Bit (CB) in Section 2. Cache Decision algorithm use the same CB as a marker to cache the content. If the CB value is one, the current router will cache the content. On the other hand, if the value of the CB is $zero$ current router will not cache the content. This kind of caching is called off-path caching, which stores the Data at a predefined location.

In this algorithm, the popularity of each chunk $LP_i(t+1)$ and threshold θ_1 are calculate by the equation that is mention in Section III-C. By applying one bit CB, the routers eliminate the Content Object without effecting the cache redundancy. In this algorithm, O_i represents i^{th} chunks of the content object.

Algorithm 3 Popularity Aware Cache Decision

```
Input: Content Object O_i
if Cache bit of O_i is 1 and LP_i(t+1) > θ_1 then
    Cache object (O_i)
    Send object by PIT list
else
    Send object by PIT list
end if
```

IV. Performance Evaluation

In this section, the performance of our proposed scheme is evaluated by using chunk-level simulator, ccnsim [15] which was developed under Omnet++ simulator. We used heterogeneous cluster size and heterogeneous cache size for simulation. Table 1 shows the parameters used in the experiments. Clients request the contents in a random manner governed by the zipf distribution. The performance of the proposed scheme and others are measured by success hit defined in (1) and hit distance. The success hit or cache hit measures the network performance which can keep on how much the routers can cache(store) the popular contents inside the network. Furthermore, the result of cache hit also reflects the server hit. If the cache hit is high, the server hit will be low. Server hit is also a measure of the traffic passing through to outside of the CCN network. Hit distance shows the number of hops that the Interest travels.

TABLE I. Parameters used in the experiments

Parameters	Values
num repos	1
replicas	1
num clients	8
λ	5
file size	20
α	1, 1.2, 1.4, 1.6, 1.8
β	0.1 to 0.9
objects	1000
CDS	LCE, LCD, PROB
RS	LRU
FS	CH,NRR , NRR1, SPR
Cache	200, 300, 400, 500, 600

We construct three groups and the network topology is depicted in Fig.5. Each cluster or group contains different number of routers. ARs connected with the users. In one cluster, the cache size of the router varies from 200 to 600 chunks. The total cache size of the whole network is 6000 chunks. In simulating other algorithms, we do not construct the groups of routers. One content provider or repository contain 1000 contents and the average size of each content is 20 chunks. Thus, the content provider contains an average of 20000 chunks. Thus, the network can cache about 30% of the total contents. Clients request the contents in a random manner governed by the zipf distribution with different parameters $1, 1.2, 1.4, 1.6$. The tuning parameter β values vary from 0.1 to 0.9.

We compared our proposed scheme with others existing schemes (three forwarding decision algorithms and three cache decision algorithms). The three forwarding decision algorithms (FS) are (1) Shortest Path Routing (SPR) [15], (2) Nearest Replica Routing (NRR) [15] and (3) Nearest Neighbor Routing 1 (NRR1) [15] and three cache decision algorithm (DS) are (1) LCE [1], (2) LCD and (3) Probabilistic in-network caching (PROB) [11]. For the cache replacement algorithm, we used Least Recently Used (LRU) for our proposal and others schemes.

Fig.6 displays the average cache hit comparison of our proposed scheme (β = 0.1) with other schemes. In here, our proposed scheme outperforms than others because our scheme stores only single copy of popular contents inside one group. The simulation result of the β from 0.1 to 0.9 for equation(1) is shown in Fig. 7 The probability of cache hit does not vary much with changing β values. Fig. 8 shows the drawback of the our proposed scheme. Our scheme can not reduce the hit distance because we used the Consistent Hashing to eliminate duplicate contents and forward the Interests. So, the router connected with the user groups can not keep the most frequently assess contents and this becomes a draw back of our scheme.

V. Conclusion

Our proposed scheme efficiently stores the contents and also it can efficiently forwards the requests. The popularity prediction assists the cache decision, in order to increase the cache hit. The effect of adding routers and removing routers can be reduced by using consistent hashing instead of modular

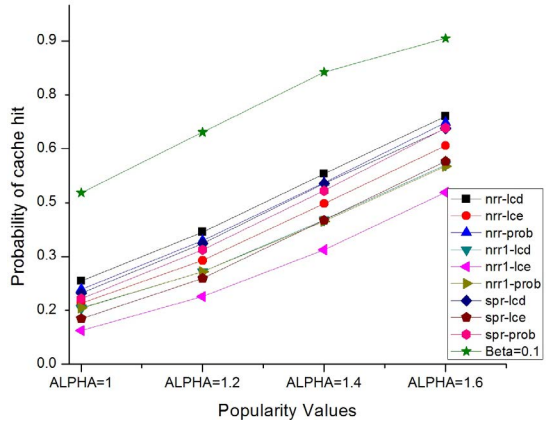

Fig. 6. Cache Hit Comparison between proposed scheme and others

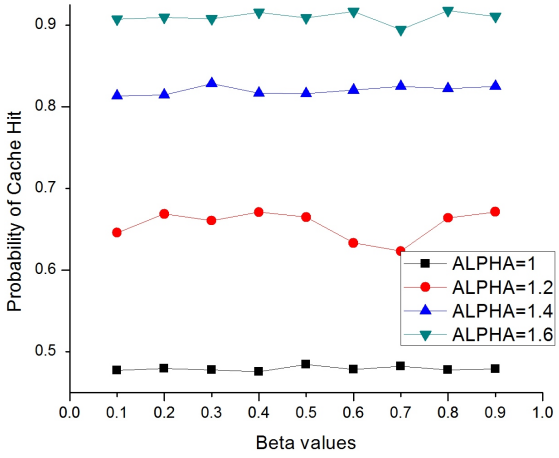

Fig. 7. Probability of Cache hit by Beta values

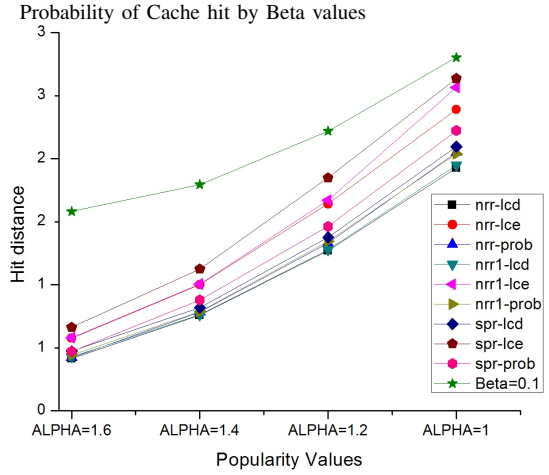

Fig. 8. Comparison of hit distance between proposed scheme and others

hashing. Also, load balance of the network is maintained by the distribution of virtual nodes. Groups of routers stored all popular Data without duplicate. By means of simulation, our scheme improved the cache hit. By the result of cache hit, we can know our proposed scheme also reduced the server hit.

ACKNOWLEDGMENT

This research was funded by the MSIP(Ministry of Science, ICT & Future Planning), Korea in the ICT R&D Program 2014. *Dr. CS Hong is the corresponding author

REFERENCES

[1] Jacobson, Van, et al. "Networking named content." Proceedings of the 5th international conference on Emerging networking experiments and technologies. ACM, 2009.

[2] Podlipnig, Stefan, and Laszlo Bszrmenyi. "A survey of web cache replacement strategies." ACM Computing Surveys (CSUR) 35.4 (2003): 374-398.

[3] Karger, David, et al. "Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web." Proceedings of the twenty-ninth annual ACM symposium on Theory of computing. ACM, 1997.

[4] DeCandia, Giuseppe, et al. "Dynamo: amazon's highly available key-value store." SOSP. Vol. 7. 2007.

[5] Li, Zhe, and Gwendal Simon. "Time-shifted tv in content centric networks: the case for cooperative in-network caching." Communications (ICC), 2011 IEEE International Conference on. IEEE, 2011.

[6] Saino, Lorenzo, Ioannis Psaras, and George Pavlou. "Hash-routing Schemes for Information Centric Networking." (2013).

[7] Wang, Sen, Jun Bi, and Jianping Wu. "Collaborative caching based on hash-routing for information-centric networking." Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM. ACM, 2013.

[8] Wang, Jason Min, Jun Zhang, and Brahim Bensaou. "Intra-AS cooperative caching for content-centric networks." Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking. ACM, 2013.

[9] Laoutaris, Nikolaos, Sofia Syntila, and Ioannis Stavrakakis. "Meta algorithms for hierarchical web caches." Performance, Computing, and Communications, 2004 IEEE International Conference on. IEEE, 2004.

[10] Guo, Shuo, Haiyong Xie, and Guangyu Shi. "Collaborative forwarding and caching in content centric networks." NETWORKING 2012. Springer Berlin Heidelberg, 2012. 41-55.

[11] Psaras, Ioannis, Wei Koong Chai, and George Pavlou. "Probabilistic in-network caching for information-centric networks." Proceedings of the second edition of the ICN workshop on Information-centric networking. ACM, 2012.

[12] Wang, Jason Min, and Brahim Bensaou. "Progressive caching in ccn." Global Communications Conference (GLOBECOM), 2012 IEEE. IEEE, 2012.

[13] Yi, Cheng, et al. "Adaptive forwarding in named data networking." ACM SIGCOMM computer communication review 42.3 (2012): 62-67.

[14] Tortelli, Michele, et al. "COBRA: Lean Intra-domain Routing in NDN." Proc. of IEEE Consumer Communications and Networking Conference, CCNC, Special Session on Research and Case Study for Designing and Deploying Information-centric Networks, Las Vegas, NV, USA. 2014.

[15] Chiocchetti, Raffaele, Dario Rossi, and Giuseppe Rossini. "ccnSim: An highly scalable CCN simulator." Communications (ICC), 2013 IEEE International Conference on. IEEE, 2013.

[16] Pinto, Henrique, Jussara M. Almeida, and Marcos A. Gonalves. "Using early view patterns to predict the popularity of youtube videos." Proceedings of the sixth ACM international conference on Web search and data mining. ACM, 2013.

[17] Szabo, Gabor, and Bernardo A. Huberman. "Predicting the popularity of online content." Communications of the ACM 53.8 (2010): 80-88.

[18] Cha, Meeyoung, et al. "Analyzing the video popularity characteristics of large-scale user generated content systems." IEEE/ACM Transactions on Networking (TON) 17.5 (2009): 1357-1370.

[19] Bernardini, Csar, Thomas Silverston, and Olivier Festor. "Mpc: Popularity-based caching strategy for content centric networks." Communications (ICC), 2013 IEEE International Conference on. IEEE, 2013.

[20] Kyi Thar, Saeed Ullah, Choong Seon Hong. "Consistent Hashing Based Cooperative Caching and Forwarding in Content Centric Network." Network Operations and Management Symposium (APNOMS), 2014 16th Asia-Pacific. IEEE, 2014.

[21] Comer, Douglas. Internetworking with TCP/IP. Page 226. Upper Saddle River, N.J.: Prentice Hall, 2000. Print.