LCD-Based on Probability in Content Centric Networking

Dang Tran Phuong[1], Tuan-Anh Le[2(✉)], Le Phong Du[3], Tuyet Anh Thi Nguyen[2], and Phuong Luu Vo[4]

[1] VNPT-NET, Ho Chi Minh City, Vietnam
tranphuong@vnpt.vn
[2] Faculty of Information Technology, Thu Dau Mot University, Thu Dau Mot, Binh Duong Province, Vietnam
{letuanh,tuyetnta}@tdmu.edu.vn
[3] Lac Hong University, Bien Hoa, Dong Nai Province, Vietnam
lpdu@tvu.edu.vn
[4] School of Computer Science and Engineering, International University - VNUHCM, Ho Chi Minh City, Vietnam
vtlphuong@hcmiu.edu.vn

Abstract. Nowadays, Content Centric Networking (CCN) could be the future Internet architecture for its advance feature: In-network caching due to cheaper to cache than to transmit contents nowadays. Besides, the popularity of content in CCN gives much challenge to researchers, but there are a few solutions for that, such as LCD (Leave Copy Down). LCD policy is known as simple and effective caching mechanism so far. This work presents a new way of cooperation between CCN nodes in caching mechanism. Our caching mechanism is an optimization of LCD that some CCN nodes can help the others for their work on caching decision policy. This cooperation method doesn't require both additional signaling packet and much computation resource to work. Experiments show that this optimization in terms of cache hit can achieve better than convention LCD does. In additional, our solution is simple and very low overhead.

Keywords: CCN · Optimization · Caching · Cooperation · Probability · LCD

1 Introduction

CCN has been nominated as a hopeful alternative to Internet connection-oriented model which use TCP/IP protocol. The principle "connection" is not be used in CCN anymore; the "content" will be central, other aspects as request and respond data, cache, store, routing or security is based on content which are given name to identify. The name of content is distinct in global scope, the named content with different sizes is split into chunks with equal size as a transmission units. The idea of CCN is to try to reuse content for many consumers' need by caching it at every CCN node. All network nodes are routers which equipped with memories for purpose of caching chunk. So that in-network caching is respected as CCN's key feature.

In CCN, the consumer shows his need to network by sending a stream of request packets for chunks of every needed content. After sending request packets, the consumer will waiting for incoming data packets that the network responds or he will resend his need again since time-out. All the request packets will propagate through node to node in network to find its suitable chunk until it hit cache at CCN node. For every request packet, the CCN node responds with a data packet - in case of hitting cache - to the requester or forward that request packet to its neighbors as it is a requester by using its forwarding policy - in case of missing cache. The CCN node use its caching policy (i.e. decision policy, replacement policy) to treat with every received data packet, these policies help it knows whatever chunk to cache to memory or to discard from the memory because of space for new one. By this way, CCN can push needed content closer to consumer than before so that can reduce network's bandwidth consumption and the content retrieval times is shorter than in TCP/IP protocol.

However, the CCN's policy (i.e. forwarding policy, decision policy and replacement policy) have been exposing many challenges for researchers. Like the importance of routing in host-to-host network paradigm which was researched well in recent decades, caching is the most important factor to CCN performance base on named content that has much attention but few studies so far. There are many studies of network caching for World Wide Web which pointed out in [2] but are not completely suitable for CCN, except one of them, LCD [2, 10] is simple and more effective than original LCE (Leave Copy Everywhere) not only in World Wide Web caching but also in CCN which [3] pointed out.

With the advances of LCD for CCN, our work in this paper proposes a novel method of caching which use LCD mechanism combines with dynamic probability. Not the same as convention LCD that always cache chunks with probability equal 1, in our work, the CCN nodes will cooperate with each other's to calculate the probability to cache chunk. Our simulations result has proved to achieve more gain to network performance than convention LCD does.

The remaining paper is structured as follows. Section 2 is a brief express of some related works. The details of this scheme and algorithm are described in Sect. 3. The simulation results are analyzed in Sect. 4 and a brief conclusion is in Sect. 5.

2 Related Work

Caching is reliable not a new topic, with many work relating with Web caching replacement policies and decision policies [1]. Relating with caching policies, as indicate in [2], much attention has been paid for cache replacement problems (more than 38 strategies were overviewed in [1]), however, the other aspect, cache decision policies, there are few studies relating with [3].

With the policy for replacement an object in cache memory, the most popular is Least Recently Used (LRU) which has been used in the CCN context [4–6] and of the more general ICN context [7–9]. Some others related with Most Recently Used (MRU) and Most Frequently Used (MFU) in ICN context [8].

For decision policies, generally the assumption is made that any new content gets always cached, these use Leave Copy Down (LCE) mechanism. As in [3], few exceptions to this rule come from the Web caching [2, 10, 12, 13] or CCN [11] contexts. In those solutions, the approach in [12] is too complexity to apply in CCN, while DEMOTE [13] is known to poorly perform on network of caches. Beside, another simple policy is considered in [11], where caching decisions are taken uniformly at random with a fixed probability FIX(P). FIX(P) can make a quite good result in caching performance of Web caching with a small probability (P approximate 0.2) in some cases. FIX(P) with small probability (0.2) may avoid many replacement errors (replacement error occurs when a higher popularity content was replaced with lower popularity in caching memory [2]) but the results of applying FIX(P) to web caching are so different depend on network topology and the amount of content or cache size. However, FIX(P) has not been applied or evaluated in CCN so far. Another decision policy that [2] introduces is MCD (Move Copy Down), with its mechanism, MCD can have a good result in web caching than LCE because of well treat to popularity contents and fewer replacement errors.

With this respect, only the Leave Copy Down (LCD) policy [2, 10] is simple enough to be worth implementing in CCN. LCD has been applied in CCN and it shown that LCD achieves much better than LCE in caching performance, especially for contents with classified popularities.

On the other hand, only a few explicit cache coordination policies (e.g., see [14] for Web, and [15] for ad hoc domain) but [3] points out that they would likely violate CCN line of speed constraint.

In direct approaches, [3, 16] study arbitrary networks of CCN caches with special attention to different caching replacement (e.g., LRU, MRU, etc.) and decision policies (e.g., LCD [1], etc.). In these works, the CCN network is considered with homogeneous cache sizes, e.g., content store of CCN nodes have the equal size.

In this work, we still focus on decision policy of CCN which use LCD in a novel way base on the idea of probability and CCN node can cooperate with each other in making a decision to cache contents.

3 Scheme Design and Algorithm

In this section introduce LCD-based on probability, called LCD-Prob. The idea of LCD-Prob is to improve cache-hit ratio of some CCN nodes with lowest performance, so that the cache-hit ratio of whole network, in average, will be better. In this scheme, with every content requested by consumer, the first node, that the request packet for each content come to, called edge node, and the others, on the path of this request packet forwarded to the repository, called core node, as shown in Fig. 1.

In imitation, when all CCN nodes are implemented with the same memory size (Content Store), we found that the core nodes usually have lower cache-hit ratios than the edge nodes in using LCD as decision policy. Since the traffic throughput at core nodes are more than at edges that lead the replacement of chunk, in limited memory, occur more frequently.

To improve the effectiveness of caching at these core nodes, they must try to retain the most popular contents in their memory. The least the memory replacement occur in core nodes, the more chance the popular contents would be retained. LCD-Prob will try to keep contents at core nodes not be changed so regularly in CCN network, so that the

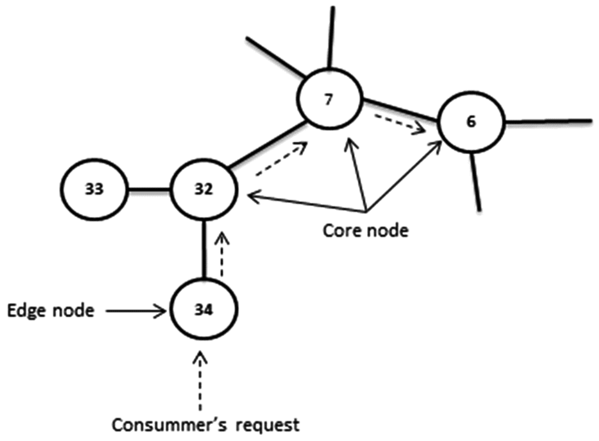

Fig. 1. Topology

decision to cache a content of LCD now is based on probability. The probability to cache is a dynamic parameter which can be calculated based on the correlation cache-hit ratio between core node and edge node. The probability denotes as: the smaller the cache-hit ratio of node is, the smaller the probability to cache a new chunk will be.

The current cache-hit ratio of edge node will be sent to next node (also core node) in every request packet that it forwards to and it will be retained in data packet on the backward path, the information is used to calculate the probability at the node which is chosen to cache the chunk by LCD's decision.

The calculation of probability is defined as follow:

n_{hit} is the number of request packet at a CCN node that hit cache;
n_{miss} is the number of request packet at a CCN node that miss cache;
C_{edge} is current cache-hit ratio of the edge node which receives request packet from consumer;
P is probability to cache a chunk in the memory of considering node;
C_{core} is current cache-hit ratio of the node which calculating probability (P) for caching a chunk;
C_x (x = core or edge) is computed by

$$C_x = \frac{n_{hit}}{n_{hit} + n_{miss}}$$

P is calculated as

$$P = MIN(1, \frac{C_{core}}{C_{edge}^k}) \tag{1}$$

where k is a constant which $k \geq 1$.

The algorithm for data packet processing at each CCN node where LCD policy decides to cache the chunk to node's memory, this node uses the information contained in data message (C_{edge}) and its information (C_{core}) to calculate the probability P. Then the CCN node will cache the chunk with probability P.

Algorithm 1: Algorithm of LCD-Prob.

```
1: Receive a data packet
2: IF (LCD policy decide to cache the chunk) DO
3:   IF($C_{edge}$ > $C_{core}$) DO
4:         $P = C_{core} / (C_{edge})^k$
5:   ELSE $P = 1.0$
6:   Generate a random number R ∈ [0,1]
7:    IF (R ≤ P) DO
8:         Cache the chunk
9:    END IF
10:  END IF
11: END IF
12: Forward data message to next node
```

When a CCN node receives a data packet, the LCD-Prob policy is used to decide to cache down this chunk or not. The same as convention LCD, for every request packet that got cache hit event at a CCN node, only the next node on the backward way (toward consumer) have chance to cache down this chunk. To cache this chunk, the CCN node must use C_{edge} (i.e., it is piggybacked in data packet) and its cache hit ratio (C_{core}) to retrieve probability P in case $C_{edge} > C_{core}$ or P will be 1, that is $P \in [0,1]$. To use P, the core node will generate a random R which $R \in [0,1]$ then compare R to P as: If $R \leq P$, the caching is successful. Oppositely, $R > P$ will cause the caching fail. The parameter k is used to adjust the change of P according to the change of C_{core}.

Finally, the data packet will be forwarded to next node toward consumer without caching down at any others.

LCD-Prob doesn't require any message packet or bandwidth to operate, it only requires a little CPU resource to calculate (1), generate a random R and compare them.

4 Evaluation

To evaluate LCD-Prob, we simulate it on CCN network by using Watts-Strogatz (WS) model [18], which can capture characteristics of Internet topology structure. Our simulation tool is implemented by C++ programming language. Consumer's requests follow Zipf distribution with the parameter $0.7 \leq \alpha \leq 1.5$. The key performance indicators are focus on average cache-hit ratio and average hop-count (i.e., total hops on the path of a request until hitting the chunk). The parameters are set in our simulation as described in Table 1.

Table 1. Parameter setting

Parameter	Value	Parameter	Value
Number of CCN node	44	Forwarding	Shortest path routing
Number of repository	12	Decision	Convention LCD
Number of consumer	24		LCD-Prob
Replacement policy	LRU	Content store	200
k	5	Number of content	5000

As shown in Figs. 2 and 3, LCD-Prob achieves better results in cache-hit ratio and hop count than convention LCD with the content following the Zipf distribution and the parameter $0.7 \leq \alpha \leq 1.5$: the network average cache-hit ratio is higher up to 16% and the network average hop count are lower 11% (Fig. 4).

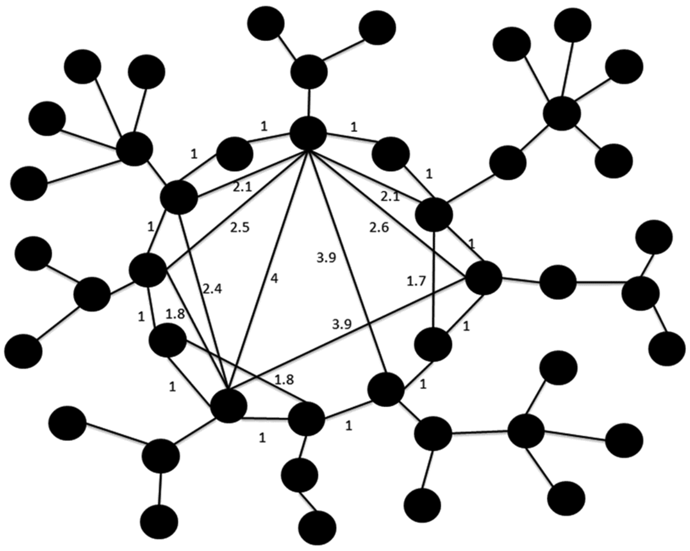

Fig. 2. Simulation topology

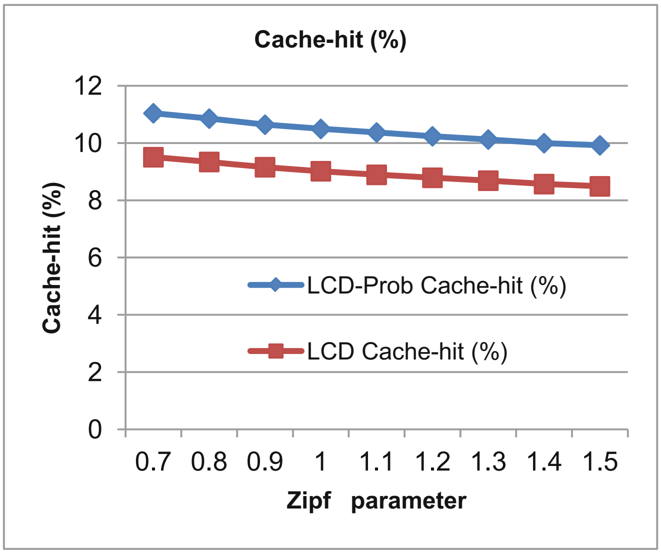

Fig. 3. Cache hit ratio (%)

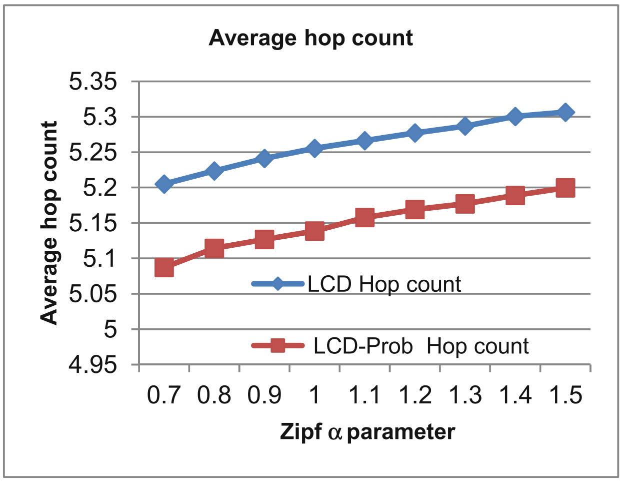

Fig. 4. Average hop count

The cache-hit ratio higher means caching performance is better and the hop count lower means LCD-Prob pushes needed content closer to consumer than LCD.

With a limit cache-hit ratio at every edge node, most of the requests will be forwarded to nearest core node to find provision server. This causes the throughput through core nodes are much higher than in edge nodes. The incoming requests are much higher but network cache size is homogeneous for all nodes will cause cache hit ratio at core nodes are smaller than at edge nodes. Moreover, LCD mechanism treats popularity content well at edge nodes but badly at core nodes, especially at the nodes next to provision server, LCD does the same as LCE.

Our mechanism can avoid that problem of LCD by caching with probability at every node especially at core nodes. The core nodes always have smaller cache-hit ratio can have smaller probability to cache a new chunk. Like the result of FIX(P) [2] has shown to us, with small probability to cache a new chunk, node CCN can reduce a lot of replacement errors would happen through caching replacement. When a node reduce this error through caching, that means it retain higher popularity contents in its memory instead of replacing them with lower popularity ones. In this case, the node' caching performance is better. With many core nodes have better performance, the network average cache hit ratio increases as we achieve in experiments.

5 Conclusions

In this paper, we propose a lightweight way to cooperate with each other on caching using statistic cache-hit ratio and without sending any additional message. Cache-hit ratio is sent to next node within request message and will be retained in data packet on backward path; CCN nodes use this info to calculate the probability to cache the chunk in combination with LCD algorithm. The proposal improves the caching performance in some nodes in network so that they have higher caching performance of network. In additional, our solution is simple and very low overhead. However, the parameter k is effected in network topology that need to be optimized to find the suitable value to achieve proposal's best result, our next work will find out this question.

Acknowledgment. This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.02-2015.36.

References

1. Podlipnig, S., Osz, L.B.: A survey of web cache replacement strategies. ACM Comput. Surv. **35**(4), 374–398 (2003)
2. Laoutaris, N., Che, H., Stavrakakis, I.: The LCD interconnection of LRU caches and its analysis. Perform. Eval. **63**(7), 609–634 (2006)
3. Rossini, G., Rossi, D.: Caching performance of content centric networks under multi-path routing (and more). Technical report, Telecom Paris- Tech (2011)
4. Psaras, I., Clegg, R.G., Landa, R., Chai, W.K., Pavlou, G.: Modelling and evaluation of CCN-caching trees. In: IFIP Networking (2011)
5. Carofiglio, G., Gallo, M., Muscariello, L., Perino, D.: Modeling data transfer in content-centric networking. In: ITC (2011)
6. Carofiglio, G., Gallo, M., Muscariello, L.: Bandwidth and storage sharing performance in information centric networking. In: ACM SIGCOMM, ICN Worskhop, pp. 1–6 (2011)
7. Choi, J., Han, J., Cho, E., Kwon, T.T., Choi, Y.: A survey on content-oriented networking for efficient content delivery. IEEE Commu. Mag. **49**(3), 121–127 (2011)
8. Katsaros, K., Xylomenos, G., Polyzos, G.C.: MultiCache: an overlay architecture for information-centric networking. Comput. Netw. 1–11 (2011)

9. Rosensweig, E.J., Kurose, J.: Breadcrumbs: efficient, best-effort content location in cache networks. In: IEEE INFOCOM (2009)
10. Laoutaris, N., Syntila, S., Stavrakakis, I.: Meta algorithms for hierarchical web caches. In: IEEE ICPCC (2004)
11. Arianfar, S., Nikander, P.: Packet-level caching for informationcentric networking. In: ACM SIGCOMM, ReArch Workshop (2010)
12. Che, H., Tung, Y., Wang, Z.: Hierarchical web caching systems: modeling, design and experimental results. IEEE J. Sel. Areas Commun. **20**(7), 1305–1314 (2002)
13. Wong, T., Ganger, G., Wilkes, J.: My cache or yours? making storage more exclusive. In: USENIX Annual Technical Conference (2002)
14. Wolman, A., Voelker, G.M., Sharma, N., Cardwell, N., Karlin, A., Levy, H.M.: On the scale and performance of cooperative web proxy caching. In: ACM SOSP, pp. 16–31 (1999)
15. Fiore, M., Mininni, F., Casetti, C., Chiasserini, C.-F.: To cache or not to cache. In: IEEE INFOCOM, pp. 235–243 (2009)
16. Rossi, D., Rossini, G.: A dive into the caching performance of content centric networking. Technical report, Telecom ParisTech (2011)
17. Jacobson, V., Smetters, D., Briggs, N., Thornton, J., Plass, M., Braynard, R.: Networking named content. In: ACM CoNEXT (2009)
18. Watts, J., Strogatz, H.: Collective dynamics of ‘small-world’ networks. Nature **393**, 440–442 (1998)