Efficient Caching and Replacement Strategy in Content Centric Network (CCN) based on Xon-Path and Hop Count

BASMAH ALOTAIBI
King Saud University
Computer Science Department
Saudi Arabia
CS.Basmah@gmail.com

SAAD ALAHMADI
King Saud University
Computer Science Department
Saudi Arabia
Salahmadi@ksu.edu.sa

Abstract: Information Centric Network (ICN) moves the internet from being host-based to become content-based. ICN provides in-network caching, named-based routing and multicast support. ICN cache the content into nodes in the network based on caching and replacement strategies. In this paper, we propose a caching and replacement strategies for content in Content-Centric Network (CCN). The caching strategy will choose the node that will be cached on based on the network topology. The proposed replacement strategy will take in its consideration the number of resources that the content has been consumed and if the content has been requested recently or not. To evaluate our proposed work, we use a ccnSim simulator, and the simulation results show that our proposed caching strategy provides more significant result than the Leave Copy Everywhere (LCE) strategy and the replacement strategy provide more significant result than the Least Recently Used (LRU) replacement strategy.

Key–Words: Information Centric Network, In-network Caching, Caching strategy, Replacement strategy.

1 Introduction

Information Centric Network (ICN) has gained attention from the research community after Van Jacobson in [1] identify the basic paradigm shift in the internet services [2]. The difference between the current internet and ICN is that ICN has the ability to use names of the content/data/information [1] instead of the content host address to reach the content. The main advantage ICN provides that it will increase the delivery speed of serves [3]. ICN moves the internet from being host-based to become content-based. The main reasoning of this shifting that is users are more interesting of the content rather than the source location [2] [4].

Nodes/Hop/Router [2] in ICN have the ability to store a copy of content that has pass through it since they are equipped with cache memory. In-network caching is one of the features provided by the ICN that will cache the traversed content based on the caching and replacement strategy has been identified in each node [5]. There are several caching and replacement strategy each one of them takes in its consideration different aspects. Different projects have been working on in ICN. Content-Centric Network is one of the ICN project [6] [7].

In this paper, we propose a caching and replacement strategy for CCN content. The proposed caching strategy aims to choose some node to cache on it and avoid caching on the whole path. The proposed replacement strategy is considered the number of resources that have been consumed in the stored content and aims to not sacrificed this content faster.

Our specific contribution includes the following: (1) Design caching strategy that varies between the nodes that will cache on it based on path length, which will reduce caching the contents in the same node. (2) Design replacement strategy that considers the resources that the content has been consuming it to reach this point and the content desired level.

This paper organized as following, Section 2 shows an overview about ICN. Related works for caching and replacement strategy are shown in Section 3. Section 4 describes our proposed scheme. Result and Evaluation are shown in Section 5. Concluding the paper in Section 6.

2 Background

Since the beginning of the internet it was running over the TCP/IP protocol, which is host-based and the information is routed for exchange based on the network address, the internet was designed for handling the communication needed when the aim of network was to share expensive and scarce resources [6] [7] [8]. The internet was not designed to handle the new requirement that generated by the enormous growth of

[1] The terms will be used interchangeably
[2] The terms will be used interchangeably

the internet, examples of some of the new requirements are security, mobility, support for scalable content distribution. This increasing number of machines led to the importance of having a solution to solve the IPs problem, which is the increasing number of machines while the IPs are limited in numbers. Information Centric Network (ICN) is one of the suggestion solution to overcome this problem [1] [8].

The internet communications are based on where the content located, while end users care about the content itself rather than its location. ICN replaces the where with what. Instead of searching where is the content located, it searches about what is the desired contents. This move the internet from host-based to content-based, thus the place of where the content is stored will not be associated with it when searching of specific content [1] [8]. ICN network is based on named content, ICN named the information at the network layer, this could deliver the data faster to the users by using in-network caching and multicast mechanisms supported by ICN. Each content has unique, persistent and location-independent name. ICN is aimed to improve the reflect of current and feature needs better than the current internet architecture. ICN provides in-network caching, named-based routing, multicast support and easy data access [6] [9].

Content Centric Network (CCN) and Named Data Network (NDN) are popular ICN projects. We will focus our talk in this paper about them. A survey about other projects in ICN can be found in [6] [7].

Content Centric Network is one of the ICN architecture. CCN start as research project at the Palo Alto Research Center (PARC) [3] [7]. There are two types of packets in CCN: interest packet and data packet. Interest packet sends by a client to request a content by its name. Data packet is the response for the interest packet with desired content. Interest packets request the contents by their names, this name is a prefix of the content name in data packet, the content will be satisfies if the name match. Name in CCN are binary object composed number of components, opaque and hierarchical [1].

CCN node has three main data structure Pending Interest Table (PIT), Forwarding Information Base (FIB), and Content Store (CS). PIT is responsible of keeping track the interest upstream toward the producer, thus the returned data packet can send downstream to the client. FIB is used to forward the interest packet to the possible producer that can have the data packet. CS is like memory buffer used to store data packet since the same data packet can be used by different user, CS have different replacement strategy and try to keep data packet as long as possible [1]. Named Data Network (NDN) is a US funded project and its predecessor to the CCN [6] [10].

Caching can be defined as having data stored temporarily on a small memory for future use. In ICN the request of content (interest packet) received by a node can be coming from a user or was forwarded by another node of the network. If the requested content (data packet) is found in the node content store, the request will be met immediately otherwise it will be forwarded towards the content producer. When the request is satisfied and return back to the user it will be cached into node content store if the node cache strategy is fit on it [11].

The in-network caching provides transparent and ubiquitous caching, to improve network resource utilization and speed up the content distribution. The caching is transparent and there is no need to specific application to cache a content. Caching in ICN are ubiquitous and there are no longer a fixed cache point (any node can be a cache) and this will make a content availability more subtle [9] [12].

Caching can be done on the delivery path called on-path cache, or at any node in the network called off-path cache. Off-path caching require additional processes and ICN projects does not have the direct capability to handle it, in contrast of the on-path caching that ICN project have the capability to handle it. In NDN to apply the off-path caching the routing information needs to upgrade. Off-path caching increase the availability of content in the whole network regardless of the delivery path used. In on-path caching the content caching is limited to the nodes on the delivery path. In this paper we will focus in on-path caching, since its supported by ICN projects [3] [13].

The advantage of in-network caching in ICN that is satisfy the requests can be done faster than retrieve data packet from original producer by caching data packet in network [6]. There are several caching and replacing strategy has been used to cache and replace the cached data in ICN in the following section we will highlight some popular caching and replacement strategy.

3 Related Work

To cache a content into cache we have the caching and replacement strategies has been used, these two strategies may complement each other [13]. Caching strategy is used to determine which content to be cached and in which node. A replacement strategy is a strategy used to determine which content is will be chosen to eliminate from the cache memory when the cache is full [7]. The need of caching and replacement strategy is that the caches are limited in term of size [13].

There are several caching and replacement strate-

gies proposed, in this section we will cover some of these strategies that used for ICN content.

3.1 Caching Strategy

Caching strategy can be based on one characteristic or several characteristics combined together to reach the caching decision. Leave copy everywhere (LCE), Leave Copy Down (LCD), Popcache and Betw are some of the caching strategy that based on one characteristic [4] [12] [14].

The default caching strategy in NDN/CCN is the LCE. LCE will leave a copy of the content in each node along the path from producer to end user. LCE can be considered as a probability strategy with caching probability equal to one in each node. LCE designed to reduce user access time to a content and minimize the frequent download from content producer. The main disadvantage of this strategy is the redundancy of caching and this strategy is causing the low utilization of cache resources [7] [12].

To reduce cache redundancy in ICN the LCD is designed. LCD caches the content only at the direct downstream node of the node that cache hit occurs on it. Popular content tend to be cached close to the end users [7] [12].

The authors in [4] propose a cache less for more approach, the content will be cached into the node that have higher probability to get cache hit. The strategy is based on the betweenness centrality (Betw) for each node, the content will be cached at the node that have a betweenness value that equal or higher than the betweenness value attached with the content. They propose also an approach for betweenness that support dynamic network (EgoBetw), nodes will calculate their betweenness based on ego network rather than the entire network. Betw obtain the best server and hop reduction ratio across different topology, while EgoBetw in nonregular topologies approximately closely the Betw.

Popcache policy is based on popularity, the popular content cached close to the user while unpopular content will be cached close to the server. Popcache achieve high performance when Zipf exponent is high. [14].

Several approaches have been proposed in [5] [15] [16] [17] and [18]. All of these approaches combine some characteristic to elect the data content to be cached. These approaches depend on several characteristic and may modify some of the popular caching strategy to apply it on specific scenario.

In [15] the probCache has been proposed.That cache the content based on probability depend on capability of paths as well as the distance between end user and producer. The probability to cache will be increased if the content get close to the end user and the cache is big in term of size. Their work decreased the cache evictions comparing with universal caching strategies. PropCache saves server hit and to hit a content in a cache its reduce the number of hops needed.

The caching policy proposed in [5] is MAx-Gain In-network Caching (MAGIC). MAGIC depends on the content popularity and hop reduction. This strategy takes in its consideration the placing gain which is the product of content popularity and number of hop from server to this node. while the replacement penalty is the minimum placing gain value in the path. To cache a content data into a node in the path, that node must have the highest local gain between the other nodes. The local gain is the difference between the placing gain in that node and the replacement penalty in the path. They take the replacement penalty in their account to reduce the caching operation. Their work shows that when the cache capacity is limited the performance is well.

In [16] the author proposes the first study on caching mechanism for IoT data in in-network cache. Where the probability of caching depends on the freshness level of data in wired network as well as the distance between producer and end users. The requester requests some level of freshness. They focus on the cost benefit of caching in-network. Cost function is depending on several factors like data freshness, level of freshness that has been requested and the multi-hop cost. The nodes dynamically change their caching probability based on cost function, the probability of caching can be increased or decreased. There are trade-off between data freshness and multi-hop cost.

In [17] probabilistic CAching STrategy for the INternet of thinGs (pCASTING) proposed a caching policy for wireless network. This cache policy is utility function based on the content and the Internet of Things (IoT) device attributes. Device attributes are the energy level of the device (remaining energy in the node) and cache occupancy. While the content attribute is the data freshness. The relationship between cache occupancy and the probability of cache is reversed, and the relation between data freshness and energy level with caching probability is positive relation. pCASTING was effective in term of energy consumption and content retrieval delay.

WAVE is a chunk-based caching strategy based on content popularity. The number of chunks that will be cached in the downstream node has been recommended by the upstream node. Number of chunk is increased exponentially with increasing number of requests. The first chunk will be cached into the first node close to the server. Each node will cache a chunks coming from its upstream node, therefore

the cache will be hop-by-hop toward the end user. WAVE have achieved the less frequent cache replacements and higher cache hit ratio than other on-demand caching schemes [18].

3.2 Replacement Strategy

Cache is filled after a certain amount of running time. Since that, a replacement strategy is needed to cache a new upcoming content [2]. Replacement strategies can be categorized based on several characteristics has been proposed in for example, Time to Live (TTL), popularity, usage time and others. These characteristics can be used by them self or combining some of them together to obtain a strategy.

The most popular replacement strategy is least recently used (LRU), least frequently used (LFU), TTL and first in first out (FIFO), each of these replacement strategy is depending on a single characteristic. FIFO is considered a simple and easy replacement strategy, that remove the first cached content when the cache is full and there is a new content need to be cached, the upcoming content will be added at the end of the cache. The shortcoming of this strategy that it does not aware about the popularity and expiration time of content, therefore a valuable content could be removed [2] [7].

LRU is simple and popular replacement strategy in ICN architecture that remove the content that has not been used in a while. This strategy cannot consider the expiration time of content. [2] [19]. The LFU replacement strategy will remove the content that has no popularity but this strategy considers as a complex replacement strategy [2].

Replacement policy that depend on more than one characteristic has been proposed in [2], [19], [20] and [21]. In [2] the authors propose a hybrid replacement strategy that combine the time and LRU. They introduce a new term called Time to Use (TTU) which is a time stamp of a content that specifies the using time for a content. Their proposed work calculates TTU for arrived content and will not save incoming content if the average request time is higher than TTU, since this content will expire before next request. The LRU will be applied once the cache is filled and there is a new incoming content need to be cached.

Least frequent recently used (LFRU) replacement policy has been proposed in [19]. LFRU will divide the cache into two non-equal parts the small part will contain the LRU contents and eliminate the LFU content from the small part. The cache network with the proposed LFRU is convergent and ergodic.

In [21] the authors propose a caching replacement mechanism based on data type. They classify data types into three categories based on data sensitivity in term of delay, data age based and demand based data.

The utility function is used to choose what data will be replaced on the CS. The data with least value of utility function will be dropped. Priority the data been stored is based on the application that has been used. Their work uses learning and reasoning technique to increase data hit ratio and reduce the time require to retrieve the data.

Adaptive Replacement Cache (ARC) is a self-tuning, low-overhead algorithm proposed in [20]. Two LRU pages list are maintained by ARC. First one maintains the recently pages that have been recently seen once. Second page maintain the high frequency pages that has been seen more than once recently. Algorithm caches only a portion of the pages on these lists. ARC try to make the two-list similar in size and balance between workload recency and frequency features.

4 Analytical Model and Proposed Scheme

In CCN the content retrieved using the following procedures:

- End user sends an interest packet to request a content by content name.
- The interest packet will be forwarded into the producer of the content.
- When an interest packet arrives into a node the node will first check its content store (CS) to find out if the content is in its cache or not.
- If the content has been found and cache hit occur the request will be satisfied and answer with data packet, the data packet will be send along the reversed path.
- If the content not in the CS the interest will be forward based on FIT and an entry in PIT will be created for each outgoing request.
- Once the content has been found it will return as data packet along the reversed path, nodes in the path will determine if it will cache the content or not.

Considering a network topology with set of nodes (routers) N among which U $\subseteq$ N is the set of content producer. Routers are connected randomly with each other and with clients. Clients are the end user, they create the interest packet. Each node j (j $\in$ N) equipped with CS that can cache until C_i conents, PIT and FIT. The caching and replacement strategies are

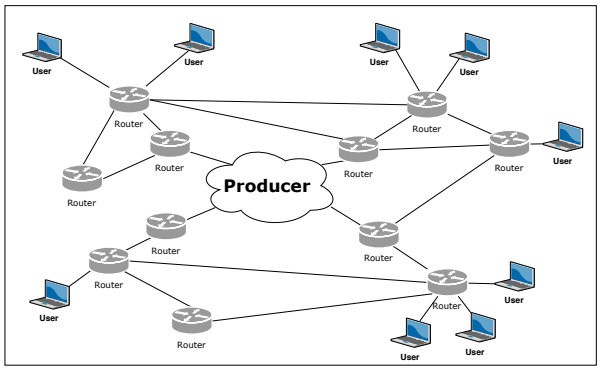

Figure 1: CCN System Model

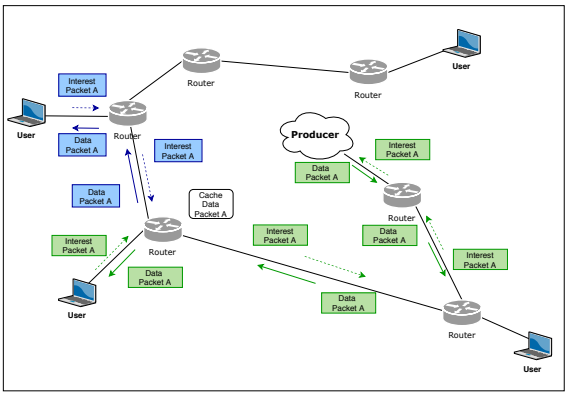

Figure 2: Example of the Xon-path Caching Strategy

the same in the whole network. Figure 1 shows an overview of the system model.

In this paper, we introduce a caching and replacing strategy. The caching strategy used is Xon-path caching. The proposed Xon path is a caching strategy that varies between the nodes that will cache on it; thus the cache will not full quickly, and different nodes will be chosen to cache on it even if the end user is the same user. For example, if the path between node Y and Z is A, B, C, D, E. and the path between W and Z is B, C, D, E. We will cache the content in the fourth node which will be node D in the first path and in the node E in the second path using the proposed caching strategy which avoids caching in the same node with different paths. The cache will be in as shown in equation (1).

$$Cache = X * i \tag{1}$$

Where the value of X>0 and $i = 1, 2, 3, ., n$. The network designer will choose the X value, and this value should be determined carefully if the value is a small number this may result with full the cache quickly and seem similar to cache on the whole path, while if the value is a high number the cache may not occur. While i is the factor that increments the X value. For example as shown in Figure 2 we have the X value = 3. Since that the nodes chosen in each path to cache on it is the third node, sixth node, ninth node and so on. The node will be chosen to be cache on it will be different in each request since the path will be changed, this will distribute the caching between nodes randomly and could result with caching some content near the end user and producer. The increase of the nodes number in the path the more caching will happen. If the end user close to the producer (less than X nodes between them) no caching is needed. The proposed strategy called Xon-path.

Algorithm 1 HLRU Replacement Strategy

```
Hops =Get the number of hop the data content travel
if Cache is empty then
    Increase cache size // (Cache content at top of the cache)
    Value = 1
    TotalValue = Value + Hops
    Set as MRU
    Set as LRU
else
    P→older = MRU // (Set the MRU as older for new content)
    MRU→ newer = P // (Set the new content as newer for MRU)
    MRU = P // (Set the new content as MRU)
    Value = MRU→value +1
    TotalValue = Value + Hops
    if Cache is full then
        Divide the cache into two non-equal size
        Take the last small size
        Take the minimum TotalValue between these content
        Delete from cache
        Reduce the cache size
    end
    Cache the element at the top of the cache
end
```

The proposed replacing strategy Hop-based Least Recently Used (HLRU) is based on the number of hops through which the data packet was traveling from the data producer until reaching this node, as well as based on if it has been requested recently or not. HLRU is considering the content desired level if it has been recently used or not and the number of resources that the content has been used to reach this

point. The proposed strategy will not quickly sacrifice the content that has traveled through many hops. The proposed replacing strategy will divide the cache into two non-equal halves when the caching eviction happens and evict the content form the small part since it will contain the contents that possible to be removed. The eviction will be based on the TotalValue value that is a combination of the number of hops with the desired level of the content.

For each node in the network, we will apply the following algorithm (Algorithm 1) when the content satisfies the caching decision. Algorithm 1 shows how the content will be stored in the cache and which content will be removed when the cache is full. The algorithm parameters are shown in Table 1.

Table 1: Algorithm Parameter

Parameter	Description
Hops	Number of hops between producer and this node.
P	Object contain the content will be cached associated with special parameter.
Value	The usage value of the content.
TotalValue	Indicate the content total value.
MRU	The most recently used content.
LRU	The least recently used content.

For each content arrive to the node and satisfy the caching decision to be cached, set the Hops to the number of hops the content travel from producer to reach this node, this value is important since if its high that imply the content has been travel for long time and consumed a lot of resources and we do not want to sacrifice it quickly. If the cache is empty and does not have any content the content will be store and recognize as MRU and LRU since its the only content in the cache. The Value will be set to one since the cache is empty, Value represent the position on the cache if it has been request recently or not comparing with other content in the cache. TotalValue will be the combination between Value and Hops. This value will reduce the probability of removing the content that still has been requested or the content has been travel a lot. If the cache is not empty then the MRU the content in the top of the cache will be the second content in the cache and the new content will be set as MRU. The Value will be set to be the Value of the new second content (the old MRU) increasing by one. If the cache is full then the last small part of the cache will be taken to choose a content from it to be evicted. The last small part will contain the LRU content. Comparing their TotalValue to choose the minimum value between them to be the evicted content. The new content will be cached at the top of the cache to be the MRU content.

When looking for data in the cache and it has been found the content will be moved to the top of the cache, set as MRU, and the TotalValue will be updated. If its already the content is MRU nothing will be changed. Otherwise the request will be forward to another node.

5 Experiments and Result

5.1 Simulation Setup

To evaluate the performance of our work we use a ccnSim simulator. CcnSim is built over the Omnet++ framework and its C++ package. CcnSim is chunk level simulator for CCN [22]. The simulation parameters are summarized in Table 2. That we have 32 clients in our simulation, using a tree for the network topology with 63 active nodes with cache size equal to 10^2. The file size is one chunk, and we have 10^3 files. The number of repositories is one and the type of content distribution is the Independent Request Model (IRM). The used forward strategy is Shortest Path Routing (SPR). The lambda value is equal to 20 and X value is 3.

Table 2: Simulation Setup

Variable	Value
Client	32
Network Topology	Tree
Active Node	63
File size	1 chunk
File in the network	10^3
Cache size	10^2
Number of repository	1
Content distribution type	IRM
Forwarding strategy	SPR
Lambda	20
X	3

5.2 Result

In our experiments we calculate the average hit ratio, average number of hop distance, average time and average number of eviction to evaluate the proposed strategies.

The hit ratio is calculated using the following equation (2).

$$HitRatio = \sum_{i}^{N} \frac{hit_i}{hit_i + Miss_i} \quad (2)$$

Where N is the total number of active nodes in the network. Hit ratio is equal to the summation of each node hit over the summation of hit and the miss. If the content is found in a node there will be a hit and if it is not found then it will be a miss. The average hit ratio calculated by dividing the hit ratio over the number of active nodes in the network that receive interest packets.

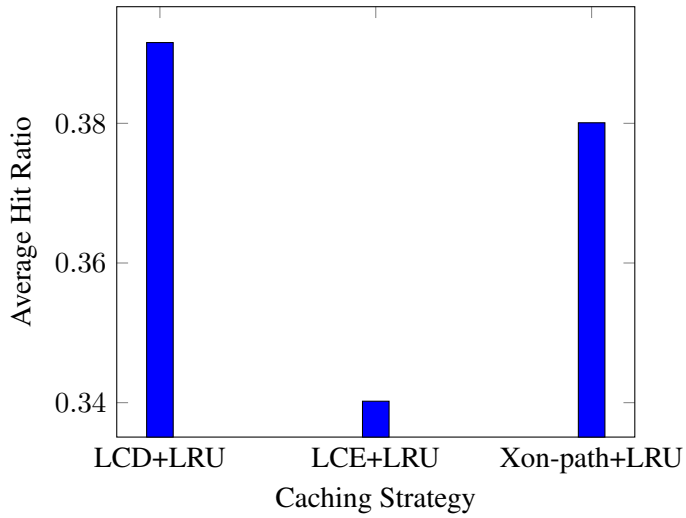

Figure 3: Average Hit Ratio for Different Caching Strategy

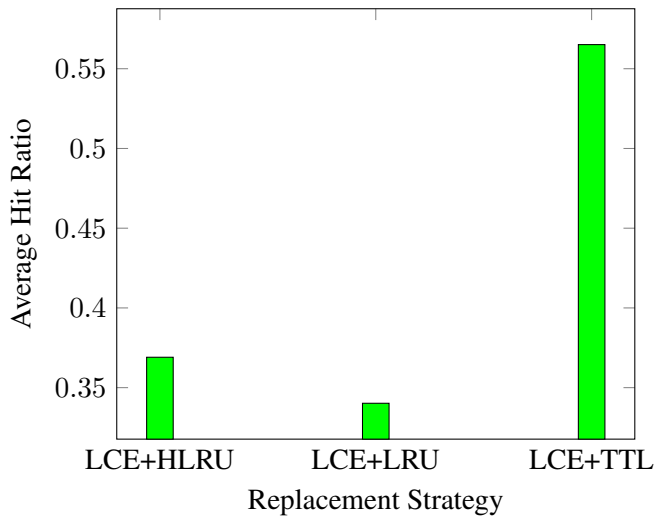

Figure 4: Average Hit Ratio for Different Replacement Strategy

We evaluate our proposed caching strategy Xon-path when the X is chosen to be three. We compare our Xon-path with LCE and LCD caching strategies when the replacement strategy is LRU. The results shown in Figure 3 shows that our proposed caching strategy provide better result than LCE, since our strategy distribute the content between nodes and each request will have different path with different number of nodes on it, which implies that different node will be chosen to be cache on in the network. While the LCE caching strategy will store the content at each node on the path which implies that the cache will be filled quickly which means that the content will be replaced and dropped rapidly. The evaluation is also done when the replacement strategy is the proposed HLRU and the result was similar with different number.

Evaluating the average hit ratio when the replacement strategies are HLRU, LRU and TTL and the caching strategy is LCE are shown in Figure 4. The results shows that our proposed HLRU is provide greater result than LRU. The result shows that the proposed replacement strategy that take in its consideration the number of hops that the content has been traveled and does not sacrificed the content that travelled for long distance quickly will increase the hit ratio even if the content was not used recently. The proposed replacement strategy HLRU provide better result than the replacement strategy that focuses only on if the content has been used recently or not. The evaluation is done also when the caching strategy is LCD and Xon-path and the result is similar with different numbers.

Average number of hop distance is calculated using the equation (3).

$$AvgHopDistance = \frac{\sum_{i}^{N} \frac{(Datasize_i * Hops_i)}{(Datasize_i + 1)}}{EndUserNumber} \quad (3)$$

The hop distance calculating is by aggregate the product of data packet size with number of hops has been travel over the data packet size +1 this aggregation will be divided by the number of end user to get the average value for the hop distance.

When evaluating the strategies using the average hop distance we are looking for the strategy that provide minimum distance between end user and the node contain the content. The average hop distance evaluation for the proposed caching strategy shows that our proposed strategy provides better result than LCE strategy. The results are shown in Figure 5. Evaluating was done also with different replacement strategies and the result provided was similar. The results shows that storing the content on several nodes on the path when the path is long will provide better result than the LCE strategy that store the content on

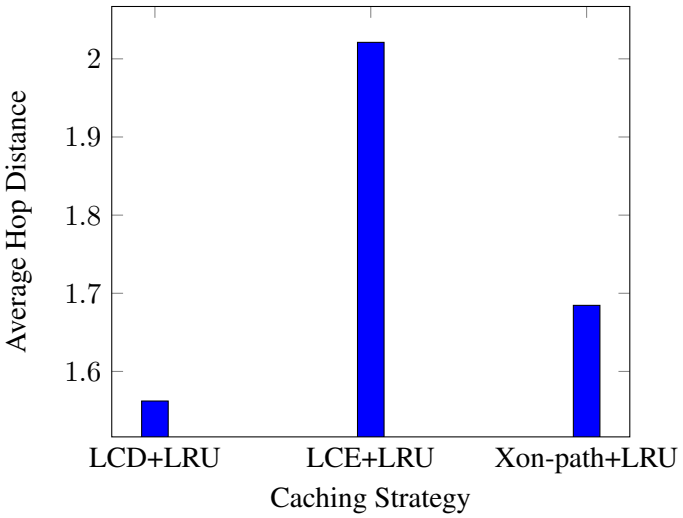

Figure 5: Average Hop Distance for Different Caching Strategy

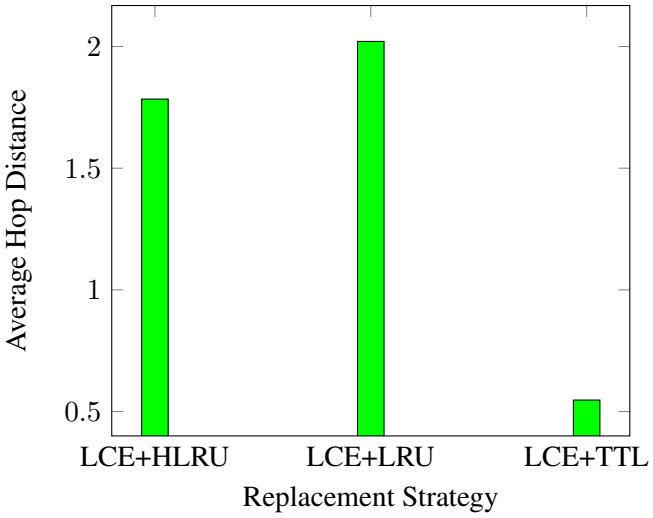

Figure 6: Average Hop Distance for Different Replacement Strategy

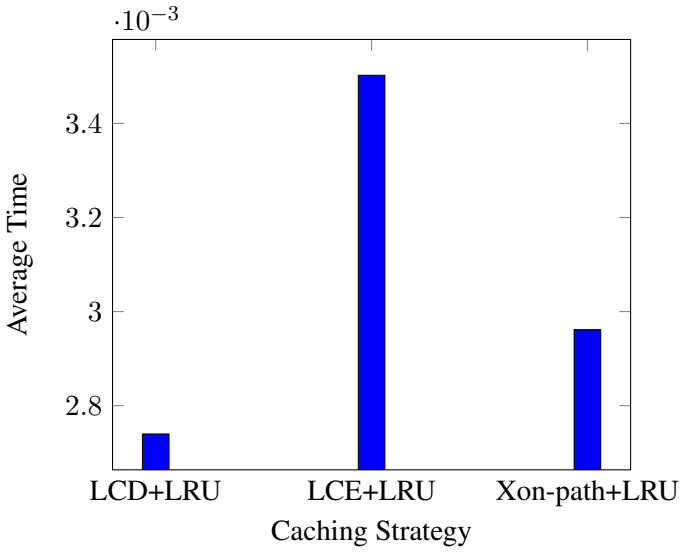

Figure 7: Average Time for Different Caching Strategy

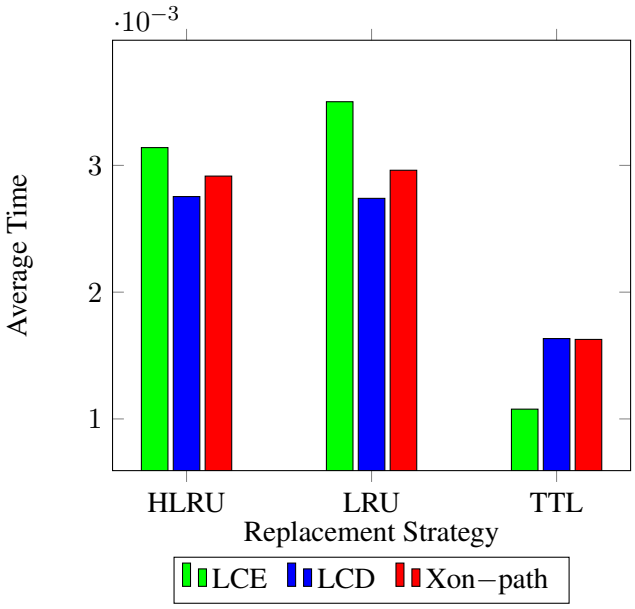

Figure 8: Average Time for Different Replacement Strategy

each node in the path which will full the cache quickly and this will result of dropping the contents quickly.

Evaluation the average hop distance with different replacement strategies and LCE caching strategy is shown in Figure 6. The results show that the proposed HLRU replacement strategy provide result better than the strategy that focus only about if the content has been used recently or not, which implies that taking in the consideration the number of hops that the content has been traveledin the replacement decision will decrease the distance needed to retrieve the required content.

The average time is the time required to complete downloading a file over the number of client. The proposed Xon-path provide a result better than caching strategies that cache on each node in the path since the time needed to retrieve the content is less as shown in Figure 7. For the HLRU its provide better result than LRU when the caching strategies are the proposed Xon-path strategy and LCE as can be seen in Figure 8.

The evaluation for average number of evictions happened shows that our proposed Xon-path caching strategy have low number of eviction compared with LCE since the LCE caching strategy will cache the contents at every node in the path which will result

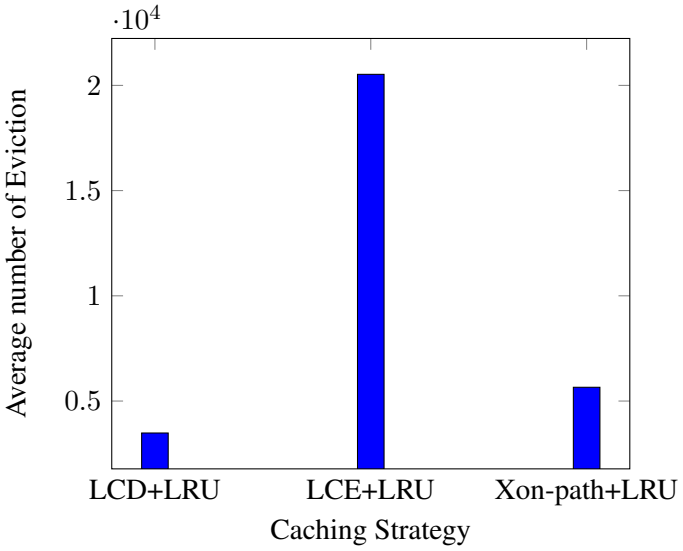

Figure 9: Average Number of Evictions for Different Caching Strategy

with filling the cache quickly. The results are shown in Figure 9.

The proposed Xon-path caching strategy increase the probability of cache hit and decrease the hop distance and time since our caching strategy take in its consideration the number of nodes in the path, that is with the increasing of the nodes numbers in the path will result with increase the number of nodes that will be cached in. The proposed Xon-path strategy decrease the eviction number by avoiding caching the content in each node in the path which is caused to full the cache quickly and increase the number of evictions.

The proposed HLRU replacement strategy provide better result than LRU replacement strategy since in our proposed HLRU replacement strategy the strategy will take in its consideration when evicting a content, the number of hop that has been travelled to not sacrificed the content that has been use a lot of resources to reach this node. Our proposed HLRU shows increasing of the hit propagability and decreasing of the hop distance and time comparing with LRU replacement strategy.

6 Conclusion

ICN become a more interesting area to the researcher at the recent time. ICN use the name of content instead of IP address when searching about a content. ICN provides in-network caching to deliver the data faster. In this paper, we propose a caching and replacement strategy for CCN. The proposed caching strategy will cache the content in several nodes based on path length. While the proposed replacement strategy is a modified version of the LRU replacement strategy that takes in its consideration the distance between producer and this node when eliciting a content. We evaluate our work using a ccnSim simulator. The results show that the proposed replacement strategy provides better result comparing with LRU replacement strategy. While the proposed caching strategy shows greater result comparing with LCE caching strategy.

References:

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, pp. 1–12, ACM, 2009.

[2] M. Bilal and S.-G. Kang, "Time aware least recent used (tlru) cache management policy in icn," in *Advanced Communication Technology (ICACT), 2014 16th International Conference on*, pp. 528–532, IEEE, 2014.

[3] I. Abdullahi, S. Arif, and S. Hassan, "Survey on caching approaches in information centric networking," *Journal of Network and Computer Applications*, vol. 56, pp. 48–59, 2015.

[4] W. K. Chai, D. He, I. Psaras, and G. Pavlou, "Cache less for more in information-centric networks," in *International Conference on Research in Networking*, pp. 27–40, Springer, 2012.

[5] J. Ren, W. Qi, C. Westphal, J. Wang, K. Lu, S. Liu, and S. Wang, "Magic: A distributed max-gain in-network caching strategy in information-centric networks," in *Computer Communications Workshops (INFOCOM WKSHPS), 2014 IEEE Conference on*, pp. 470–475, IEEE, 2014.

[6] G. Xylomenos, C. N. Ververidis, V. A. Siris, N. Fotiou, C. Tsilopoulos, X. Vasilakos, K. V. Katsaros, and G. C. Polyzos, "A survey of information-centric networking research," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 2, pp. 1024–1049, 2014.

[7] H. Jin, D. Xu, C. Zhao, and D. Liang, "Information-centric mobile caching network frameworks and caching optimization: a survey," *EURASIP Journal on Wireless Communications and Networking*, vol. 2017, no. 1, p. 33, 2017.

[8] A. V. Vasilakos, Z. Li, G. Simon, and W. You, "Information centric network: Research challenges and opportunities," *Journal of Network and Computer Applications*, vol. 52, pp. 1–10, 2015.

[9] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, K. Drira, and S. AlAhmadi, "Cache freshness in named data networking for the internet of things," *The Computer Journal*, 2018.

[10] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, *et al.*, "Named data networking," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 66–73, 2014.

[11] A. Ghodsi, S. Shenker, T. Koponen, A. Singla, B. Raghavan, and J. Wilcox, "Information-centric networking: seeing the forest for the trees," in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, p. 1, ACM, 2011.

[12] G. Zhang, Y. Li, and T. Lin, "Caching in information centric networking: A survey," *Computer Networks*, vol. 57, no. 16, pp. 3128–3141, 2013.

[13] A. Ioannou and S. Weber, "A survey of caching policies and forwarding mechanisms in information-centric networking," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2847–2886, 2016.

[14] K. Suksomboon, S. Tarnoi, Y. Ji, M. Koibuchi, K. Fukuda, S. Abe, N. Motonori, M. Aoki, S. Urushidani, and S. Yamada, "Popcache: Cache more or less based on content popularity for information-centric networking," in *Local Computer Networks (LCN), 2013 IEEE 38th Conference on*, pp. 236–243, IEEE, 2013.

[15] I. Psaras, W. K. Chai, and G. Pavlou, "Probabilistic in-network caching for information-centric networks," in *Proceedings of the second edition of the ICN workshop on Information-centric networking*, pp. 55–60, ACM, 2012.

[16] S. Vural, P. Navaratnam, N. Wang, C. Wang, L. Dong, and R. Tafazolli, "In-network caching of internet-of-things data," in *Communications (ICC), 2014 IEEE International Conference on*, pp. 3185–3190, IEEE, 2014.

[17] M. A. Hail, M. Amadeo, A. Molinaro, and S. Fischer, "Caching in named data networking for the wireless internet of things," in *Recent Advances in Internet of Things (RIoT), 2015 International Conference on*, pp. 1–6, IEEE, 2015.

[18] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, "Wave: Popularity-based and collaborative in-network caching for content-oriented networks," in *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on*, pp. 316–321, IEEE, 2012.

[19] M. Bilal and S.-G. Kang, "A cache management scheme for efficient content eviction and replication in cache networks," *IEEE Access*, vol. 5, pp. 1692–1701, 2017.

[20] N. Megiddo and D. S. Modha, "Outperforming lru with an adaptive replacement cache algorithm," *Computer*, vol. 37, no. 4, pp. 58–65, 2004.

[21] F. Al-Turjman, "Cognitive caching for the future sensors in fog networking," *Pervasive and Mobile Computing*, vol. 42, pp. 317–334, 2017.

[22] R. Chiocchetti, D. Rossi, and G. Rossini, "ccnsim: An highly scalable ccn simulator," in *Communications (ICC), 2013 IEEE International Conference on*, pp. 2309–2314, IEEE, 2013.