On-Path Collaborative In-Network Caching for Information-Centric Networks

Lemei Huang[§], Yu Guan[§‡], Xinggong Zhang[§†‡] and Zongming Guo[§†‡*]
[§]Institute of Computer Science Technology, Peking University, China, 100871
[†]Cooperative Medianet Innovation Center, Shanghai, China
[‡]PKU-UCLA Joint Research Institute in Science and Engineering
{milliele, shanxigy, zhangxg, guozongming}@pku.edu.cn

***Abstract*—It is a challenge for Information-Centric Network (ICN) that how to maximize the utilization of network-embedded cache. On-path collaboration is an efficient way to reduce access latency and cache redundancy. All nodes on ICN routing path work together to achieve higher performance than individual cache, and lower collaboration overhead than regional collaborative cache. In this paper, we formulate the traffic cost minimization problem in on-path collaborative caching, and give some insights of it. A caching utility function is defined with respect to content popularity and distance to content cache or source, in order to evaluate the payoff of caching an item at one of the on-path collaborative nodes. An on-path collaborative caching scheme UtilCache is proposed, which keeps ICN cache decision procedure unchanged but replaces items by the proposed policy LCU (Least Caching Utility). LCU evicts the items with least caching utility calculated by the caching utility function. The collaborative messages are piggybacked by data packets with low communication overhead. Compared with state-of-the-art caching schemes, UtilCache yields better performance in cache hit ratio, low latency and low communication overhead. The experimental results demonstrate that the proposed UtilCache achieves up to 50% latency reduction compared with individual caching, and less overhead compared with other collaborative caching. It also validates the implementation of UtilCache is very easy to be seamlessly integrated in current ICN framework.**

I. Introduction

Information-Centric Network (ICN) is proposed as future Internet architecture to achieve efficient content dissemination. *In-network caching* plays an essential part in ICN. Routers in the network are equipped with a cache eligible for caching content items. If a content request is served in an en-route router (i.e. the request hit the router's cache) before it arrives at source server, latency of content dissemination reduces and bandwidths of upstream links are saved. Though there are previous works in web caching (e.g. [1], [2]) for reference, in-network caching is different from web caching fundamentally and demands further research. In-network caching in ICN can be summarized in two questions: Which item should be cached? How can content requests access cached items?

The former refers to the *content placement* issue. A router can decide content placement on its own or in coordination with other routers. The latter refers to the *request forwarding* issue. Content requests can be forwarded on forwarding path decided in control plane of ICN architecture (*e.g.* FIB in NDN [3], referred as default forwarding path hereafter), or in a unique way defined by specific caching mechanism. The second alternative in both issues gives better performance in latency or storage consumption but higher overhead. In fact, there has long been a trade-off between extra overhead and optimized performance in ICN in-network caching.

Based on the above, in-network caching in ICN is classified into: *individual caching*, *on-path collaborative caching* and *regional collaborative caching*. In individual caching, each router makes independent content placement decisions and forwards content requests on default forwarding path. Individual caching is simplest without any coordination overhead, whereas achieves least satisfactory performance in cache hit ratio and latency due to lacking extra caching information of other routers. In regional collaborative caching, routers decide content placement coordinating with other routers in the network and forwarding path of content requests is possibly redesigned. Shifts in both content placement and request forwarding optimize latency or storage consumption, but introduce extra coordination overhead. On-path collaborative caching is a kind of caching where routers coordinate with en-route routers on the path to serving router and forward content requests on default forwarding path. It seems to be a compromise that achieves higher performance than individual caching and less coordination overhead than regional collaborative caching.

The motivation of our work is to ameliorate content download rate in ICN, which means to minimize the latency. One of the important ways to reduce latency is to reduce traffic cost. In this paper, we formulate the *traffic cost minimization* problem in on-path collaborative caching for ICN, and propose an on-path collaborative caching scheme **UtilCache** to address this problem. In UtilCache, routers coordinates with on-path routers to calculate the caching utility of content items, i.e. how beneficial in traffic cost reduction caching the item is, and evicts the least utilitarian ones.

Our main contribution of this paper can be summarized as:

- We formalize the *traffic cost minimization* problem in on-path collaborative caching, given local content request frequency at each router and default forwarding path.
- To measure the payoff of caching a content item at a router, we derive from the formulation a caching utility

This work was supported by National Natural Science Foundation of China under contract No. 61471009, and Culture Development Funding under [2016]288

* Corresponding author. E-mail:guozongming@pku.edu.cn

978-1-5386-2784-6/17/$31.00 ©2017 IEEE

function concerning popularity of the item and distance from the router to serving router (can be either routers with cache or source server).

- We present an on-path collaborative caching scheme UtilCache to address the traffic cost minimization problem based on the main target to attain higher caching utility, and compare UtilCache with other caching schemes based on simulation experiments. UtilCache yields higher cache hit ratio and lower latency than all benchmark on-path collaborative caching schemes.

The rest of our paper is organized as follows: Section II discusses the classification of collaborative caching and state of the art in ICN collaborative caching. In section III, we deal with the *traffic cost minimization* problem in ICN on-path collaborative caching. Section III-A proposes the formulation of the problem, section III-B derives a caching utility function from the formulation to judge how caching an item at a router benefits, and section III-C presents the caching scheme UtilCache. We evaluate the performance of our scheme in section IV, and eventually conclude in section V.

II. Related Work

Simple as it is, individual caching achieves poor performance. The focus of recent research in ICN in-network caching falls on collaborative caching. In this section, we discuss about collaborative caching schemes: *on-path* and *regional* collaborative caching schemes.

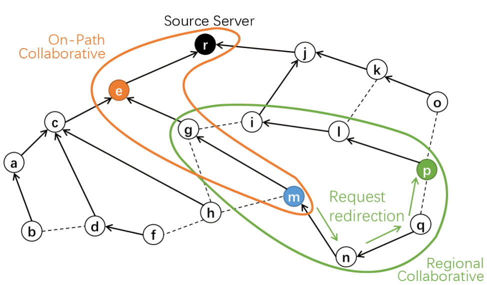


Fig. 1. **On-path and Regional collaborative caching**: r is the source server of a given item k. Default forwarding path is depicted by arrows. Routers involved in coordination in both on-path and regional collaborative caching are encircled respectively. To exchange information among routers in both circles, regional collaborative caching needs extra communication overhead. For example, information of i is unable to be passed to m following default forwarding path and communication between (g, i) or (q, p) is necessary. After coordination, content is likely to be cached in e in on-path caching (p in regional caching, to where content requests need redirection).

In regional collaborative caching, router coordinates with other routers in a certain region (entirety or part of the network) to make caching decisions. In most cases, the coordination necessitates extra information exchange among routers in the region. After collaboration, contents are possibly cached in any router of this region, for which request forwarding path always needs redesigning to redirect content requests to cached copies of the content item. Regional collaborative caching inherently gives the best optimization, but the accompanied coordination overhead limits its scalability and efficiency. Hash-routing schemes in [4], greedy heuristic method to solve intra-AS cooperative redundancy elimination problem (CRE-P) [5], and Multi-hop Neighborhood Collaborative Caching (MuNCC) scheme [6] fall into this category.

In on-path collaborative caching, routers involved in coordination are narrowed down to those on default forwarding path. Since both content requests and data packets are disseminated on default forwarding path, coordination information can be carried by them without extra overhead and there is no need to redesigned the forwarding path. Fig. 1 exhibits the difference between on-path and regional collaborative caching.

Several meta algorithms for hierarchical web caching are proposed in [7], from which Leave Copy Down (LCD), Move Copy Down (MCD) can be applied directly to on-path caching when data packets are forwarded downstream, improving latency and caching efficiency to some extent. Probabilistic on-path caching schemes are proposed as well, in which on-path routers cache contents with probability p. In ProbCache [8], probability p concerns cache capacity and distance from serving router. Content popularity, however, is another significant factor should be taken into account in traffic cost reduction. PopCache [9] improves the cache hit ratio and get lower latency than ProbCache by adding content popularity into probability calculation as a factor. However, in PopCache, each router needs global downloading statistics to calculate cache probability, which is not very practicable.

In [9], content placement issue consists of cache decision and cache replacement. We notice LCD, MCD, ProbCache and PopCache are all distinguished in content decision policies. In the following section, after formulating the *traffic cost minimization* problem, we are going to present a non-probabilistic caching scheme with distinctive content replacement policy. The scheme considers content popularity as well, but global downloading statistics are no longer necessary for each router.

III. Collaborative On-Path Traffic Cost Minimization

In this section, we start by formalizing the *traffic cost minimization* problem in ICN on-path collaborative caching. After deriving a caching utility function to evaluate the payoff of caching a content item at a router from the formultion, we present a caching scheme to address this problem based on the idea of attaining higher caching utility.

A. Problem Formulation

As is mentioned above, the *traffic cost minimization* problem aims at optimizing the content distribution to alleviating total traffic cost with limited storage. In on-path caching, the content distribution optimization involves routers on the default forwarding path.

For simplification, we make four assumptions:

1) Although a content item possibly has multiple replicas in many content providers, we suppose that each router will choose only one nearest content provider from eligible ones as source server and forward the content requests for this item on the shortest path to designated source server. Therefore, routers on this shortest path choose

the designated content provider as its source server for this item as well. How to choose the nearest content provider and calculate the shortest path is designed in the control plane in ICN architecture.

2) Contents in the network exist in chunks with unified size (let it be C). Each chunk corresponds to a content request. Thus, each chunk can be considered as an item.
3) Content request and corresponding data packet travel through the same path.
4) Content requests are much smaller than data packets, for which only traffic cost caused by data packets is taken into account.

We use a directed graph $G = \langle \boldsymbol{N}, \boldsymbol{E} \rangle$ to denote the network topology, where node set $\boldsymbol{N}$ represents routers and edge set $\boldsymbol{E}$ refers to the links between them. $\langle i, j \rangle \in \boldsymbol{E}$ means data can be transferred from node i to j. B_i denotes the cache size of node $i \in \boldsymbol{N}$ (in units of chunks). For each item $k \in \boldsymbol{K}$, as is one of the characteristics of on-path collaborative caching, there is a source server storing item k. Let $\boldsymbol{S}$ be the set of source servers for all $k \in \boldsymbol{K}$. Content requests for k are forwarded to source server on default forwarding path.

Given item k, according to assumption 1, there is one and only one next hop for the unsatisfied content requests at each node to be forwarded to. Therefore, a tree ($T^k = \langle \boldsymbol{N}, \boldsymbol{P^k} \rangle$) rooted at source server of k can be derived from default forwarding path (Fig. 2(a)), where $\boldsymbol{P^k}$ is a subset of $\boldsymbol{E}$. For $\langle i, j \rangle \in \boldsymbol{P^k}$, the parent node j of node i is the aforementioned unique next hop. Therefore, $\boldsymbol{P^k}$ represents the *default forwarding path of* k. All unsatisfied content requests for k at a node i will only be forwarded to its parent node in T^k, similarly the content returning from it as well.

Each router is an agent of several local clients. At node i, the average arrival rate of content requests for item k from local clients is denoted by l_i^k. Let $X_i^k \in \{0, 1\}$ be the storage decision for item k at node i, while $X_i^k = 1$ indicates node i caches item k. The content request rate for item k from node i to node j is denoted by f_{ij}^k, which means there are an average of f_{ij}^k content requests for k forwarded from i to j per unit time. For each node $i \in \boldsymbol{N}$, it should be noticed that content requests for k comes from only neighbors which are the children nodes of k in T^k. Thereby $\sum_{j:\langle j,i \rangle \in \boldsymbol{P^k}} f_{ji}^k$ refers to the arrival rate of content requests from neighbors for k at node i. Thus, the total query rate for item k at node i is $l_i^k + \sum_{j:\langle j,i \rangle \in \boldsymbol{P^k}} f_{ji}^k$, which can be regarded as the popularity of k at node i (denoted by p_i^k). Only when cache miss occurs are all content requests for k forwarded to and only to the next hop to source server (i.e. parent node). For all $k \in \boldsymbol{K}$ and $\langle i, j \rangle \in \boldsymbol{E}$, f_{ij}^k is calculated as:

$$f_{ij}^k = \begin{cases} (l_i^k + \sum_{v:\langle v,i \rangle \in \boldsymbol{P^k}} f_{vi}^k)(1 - X_i^k) & \text{, if } \langle i, j \rangle \in \boldsymbol{P^k} \\ 0 & \text{, otherwise} \end{cases} \tag{1}$$

Since there is one and only one parent node for each node $i \in \boldsymbol{N}$ (except root node) in T^k, we have:

$$\sum_{j:\langle i,j \rangle \in \boldsymbol{E}} f_{ij}^k = p_i^k (1 - X_i^k) \tag{2}$$

Content requests for item k introduce a bandwidth cost of $f_{ij}^k C$ at link $\langle j, i \rangle$, where C is the size of item k (assumption 2). This is because, according to assumption 3, each content request forwarded upstream traversing link $\langle i, j \rangle$ corresponds to a data packet in turn traversing $\langle j, i \rangle$ downstream. The *traffic cost minimization* problem in ICN on-path caching, reffered as **TCM-OP** (Traffic Cost Minimization On-Path) can be formulated as follows:

$$\min \quad \sum_{k \in \boldsymbol{K}} \sum_{\langle i,j \rangle \in \boldsymbol{E}} f_{ij}^k C \tag{3}$$

$$\text{s.t.} \quad B_i - \sum_{k \in \boldsymbol{K}} X_i^k \geq 0, \quad \forall i \in \boldsymbol{N}, k \in \boldsymbol{K} \tag{4}$$

$$X_i^k = 1, \qquad \forall i \in \boldsymbol{S}, k \in \boldsymbol{K} \tag{5}$$

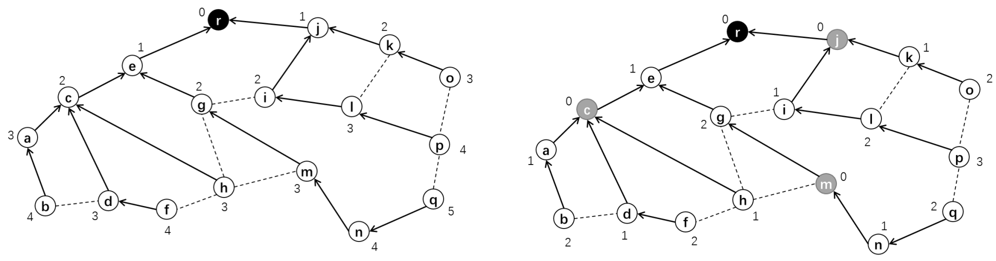
$$X_i^k = \{0, 1\}, \qquad \forall i \in \boldsymbol{N}, k \in \boldsymbol{K} \tag{6}$$

The objective function (3) refers to total traffic cost in the entirety. It can be equivalently expressed by $\sum_{k \in \boldsymbol{K}} \sum_{i \in \boldsymbol{N}} p_i^k (1 - X_i^k) C$. Constraint (4) is the storage capacity constraint. Constraint (5) indicates source server designated in control plane must store the corresponding item.

B. Caching Utility Function

TCM-OP is a problem unable to be solved in polynomial time. Pondering over the TCM-OP model, we figure out that the following intuitive observations can be demonstrated in our formulation as well:

1) **The more replicas of an item are cached, the less traffic cost.** This is because $\sum_{k \in \boldsymbol{K}} \sum_{\langle i,j \rangle \in \boldsymbol{E}} f_{ij}^k C = \sum_{k \in \boldsymbol{K}} \sum_{i \in \boldsymbol{N}} p_i^k (1 - X_i^k) C$ reduces when any X_i^k changes from 0 to 1 with p_i^k and C non-negative.
2) **It may be advantageous to cache those popular items, because the frequent requests for them bring about frequent traffic cost reduction.** Suppose $p_i^{k_1} < p_i^{k_2}$ while $X_i^{k_1} = 0$ and $X_i^{k_2} = 0$. The objective function (3) will be cut down by $p_i^{k_1} C$ when k_1 is cached, and by $p_i^{k_2} C$ when k_2 is cached. It's obvious caching the more popular one achieves less traffic cost.
3) **Consider a path between the serving node and a content consumer. The content cached more close to consumer saves more traffic cost.** Here serving node can be both source server and en-route node with cache. Let s be the serving node and t the consumer. i is any node on the path and $i \neq s$. Notice that $X_j^k = 0, \forall j \neq s$ and $X_j^k = 1, j = s$, because j will be the serving node if $X_j^k = 1$ and $j \neq s$. When i caches item k and X_i^k changes from 0 to 1, $\sum_{j:\langle i,j \rangle \in \boldsymbol{E}} f_{ij}^k = p_i^k (1 - X_i^k)$ changes from p_i^k to 0. For any upstream node v of i, $\sum_{v:\langle v,j \rangle \in \boldsymbol{E}} f_{vj}^k$



(a) No replica in the network

(b) $\{r, c, j, m\}$ store item k

Fig. 2. **Calculation of** D_i^k: r is the source server of item k

reduces by p_i^k. The objective function 3 thus reduces by $np_i^k C$, where n is the number of nodes on path between i and s. The nearer content is cached from consumer, the higher n, which leads to lower traffic cost.

According to observation 2, 3, we notice that the advantage in regard to traffic cost of caching an item at a node depends largely on *popularity* of the item and *distance* from the node to serving node (can be either source server or any on-path node with cache). We use D_i^k to denote the distance from the serving node to node i (in units of hops). In most cases, the serving node is the nearest ancestor in T^k caching item k on path between node i and source server. Given the item k:

Distance D_i^k**:** The distance between i and j, where j is nearest upstream (ancestor) node from i with $X_j^k = 1$.

For example, $D_h^k = 3$ in Fig. 2(a) and $D_h^k = 1$ in Fig. 2(b).

It is easy to prove the objective function will be cut down by $D_i^k p_i^k C$ when node i caches item k, which means the total reduced traffic cost caused by caching item k at node i is $D_i^k p_i^k C$. When content items are with different sizes, caching an item with lower size C and higher product of distance D and popularity p possibly achieves the same value of reduced traffic cost as caching that with higher size and lower product of D and p. It is obvious that caching the smaller one is better as it spares cache space for other content items. That is to say, caching items with higher average reduced traffic cost per unit storage is more utilitarian at a node. Therefore, we drive from TCM-OP a *caching utility function* as:

$$U_i^k = \frac{D_i^k p_i^k C}{C} = D_i^k p_i^k \tag{7}$$

Caching utility function U (7) evaluates the caching utility of item k at node i in TCM-OP. U offers us a new perspective to deal with TCM-OP: attaining higher caching utility as possible. This perspective leads to our caching scheme in Sec. III-C.

C. Caching Scheme

In this section, we present a caching scheme **UtilCache** to address TCM-OP problem, described in Tab. I. It is based on the main idea of achieving higher caching utility.

UtilCache belongs to on-path collaborative caching. Each router coordinates with other routers on path to serving router, and content requests are forwarded on the default forwarding path. In UtilCache, routers decide to cache every content passing by according to observation 1.

TABLE I
FRAMEWORK OF UTILCACHE

Content Placement	**Content Decision**	Cache every content
	Contetnt Replacement	Least Caching Utility (LCU)
Request Forwarding		Default forwarding path

To attain higher caching utility, routers simply do content replacement with policy LCU (Least Caching Utility) prioritizing items by caching utility, similar to LRU policy where items are prioritized by last-use time and LFU by use frequency. When cache overflow occurs, router calculate the caching utility ($U = Dp$) for each cached content item and evicts those items with lowest caching utility, therefore content items with higher caching utility stays longer in cache.

In practice of UtilCache, a *Time Since Birth* (TSB) field is added to data packet header to measure D, as it does in ProbCache [8]. Similar to TTL field in IP packets, every router data packet passes by increases TSB by one. Popularity p can be obtained by counting and recording how much content requests (from both local clients and other routers) for each item arrives in a certain time interval at each router and updating when next interval comes.

The idea of collaboration is embodied in the distance D. By updating TSB in returned data packet, upstream routers deliver the information of *how far you are from serving router* to the current router, without introducing extra communication overhead. Thus, contents are more likely to be cached at a router nearer content consumer in this coordination. UtilCache is a simple, practical greedy on-path collaborative caching scheme to deal with the traffic cost reduction problem, endeavoring to give an approximate solution of TCM-OP problem.

IV. EVALUATION

A. Experiment Setup

We use *Icarus* [11], a simulator offering flow-level simulations, to evaluate the performance of each caching schemes: Leave Copy Everywhere (LCE) [7], Symmetric Hash Routing (HR Symm) [4], ProbCache [8], PopCache [9] and UtilCache.

Routers in LCE simply decide to cache every item and do content replacement on their own, for which LCE can be regarded as individual caching. HR Symm is a hash-routing scheme with highest cache hit ratio among five hash-routing schemes proposed in [4]. We choose it as representative of regional collaborative caching schemes. ProbCache, PopCache and UtilCache are all on-path collaborative caching schemes. ProbCache is frequently used as benchmark caching scheme in most of the evaluations, while PopCache considers the impact of content popularity on traffic cost minimizaion problem.

The simulation environment is described in Tab. II.

TABLE II
SIMULATION ENVIRONMENT

Cache	**Size**	Constant, uniformed
	Replacement	LCU: for UtilCache LRU: others
Content	**Number**	3×10^5
	Popularity	Zipf Distribution
Content Request	**Number**	Total: 12×10^5 6×10^5 for cache warm-up 6×10^5 for data tracing
	Rate	120 requests per second
	Distribution	Poisson Distribution

We evaluate the performance of caching schemes with two metrics: *cache hit ratio* and *latency*. Cache hit ratio refers to the proportion of content requests hitting router's cache before arriving at source server while latency refers to the time interval between when content request is send and data packet arrives. The former reflects the server load and the latter indicates the traffic cost. Higher cache hit ratio brings about lower server load, and lower latency means lower traffic cost.

Simulation scenarios are altered in terms of network topology, Zipf exponent α and cache to population ratio C. Zipf exponent α indicates the skewness of popularity distribution. When α increases, the popular items become further popular and unpopular ones less. Cache to population ratio, introduced in [4], shows the proportion of total cache size to total content size. We execute the experiments on four real topologies: GEANT [12], WIDE [13], GARR [14] and Tiscali [15], with Zipf exponent α circumscribed in $[0.6, 1.4]$ and cache to population ratio C in $[0.2\%, 5\%]$.

B. Effect of Various Time Interval

To obtain the popularity of each item, each router counts and records the number of content requests for this item in every time interval. When a new interval begins, router regards the number of content requests for an item in last interval as the popularity of it, which is the *popularity update* procedure.

Choosing proper time interval makes great difference in popularity update procedure. If the interval is too small and popularity updated frequently, the approximated popularity is inaccurate (usually a tiny value even can be 0). In this case, UtilCache degenerates into individual caching with simple cache replacement policy such as FIFO, because caching utility U is always 0 when $p = 0$. If the interval is too large and popularity updated untimely, routers are likely to cache

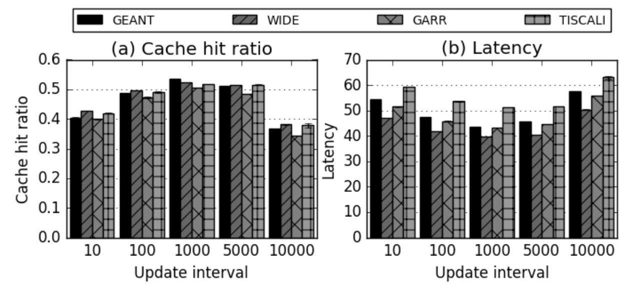


Fig. 3. Performance of UtilCache in different time interval with: request generation rate 120 requests/sec, $\alpha = 1.0$ and $C = 1\%$

newly unpopular items, taking their previous high popularity for actual popularity, which leads to lower caching utility.

Fig. 3 exhibits the performance of UtilCache with time interval tuned from 10 seconds to 10000 seconds under a request generation rate of 120 requests/sec. We notice that trends in performance are similar on different network topologies in accordance with the discussion above. Hereafter we choose time interval of 1000 seconds in our subsequent experiments. Intuitively we suppose the choice of time interval is related to request generation rate, which needs further research.

C. Performance of UtilCache

Now we compare the effectiveness of UtilCache with individual caching scheme LCE, regional collaborative caching scheme HR Symm, and on-path collaborative caching schemes ProbCache, PopCache. Based on the investigation before, we use 1000 second as the popularity update interval of UtilCache.

PopCache does not achieve its expectant performance in our simulation. We figure out several possible reasons to account for this: (1) In [9], α is circumscribed into $[1.2, 2.4]$ which higher than that in our experiments. (2) In [9], items are likely to have different sizes while we assume item sizes are the same. (3) The simulation of [9] is implemented on specific network topology: cascading and binary-tree topologies.

Fig. 4 shows the performance of each scheme in terms of cache hit ratio and content retrieval latency, when Zipf exponent α varies from 0.6 to 1.4 with cache to population ratio fixed at 1%. In regard to cache hit ratio, UtilCache yields best performance when α is small. With α increasing, UtilCache is outperformed by HR Symm, the regional collaborative caching scheme. Although HR Symm attains higher cache hit ratio, it has the inherent drawbacks as a regional collaborative caching scheme. Firstly, the way used in HR Symm to avoid the high coordination overhead is simply installing a hashing function at each router, which makes it not adaptive enough to the network changes (e.g. the change of cache availability and network topology). Secondly, the redirection of content requests and data packets possibly disables them to travel the default forwarding path (usually the shortest path), which increases the latency. In fact, UtilCache yields best latency among the five caching schemes. This is because in UtilCache router evicts the less utilitarian items to attain higher caching utility and the caching utility is bound up with traffic cost.

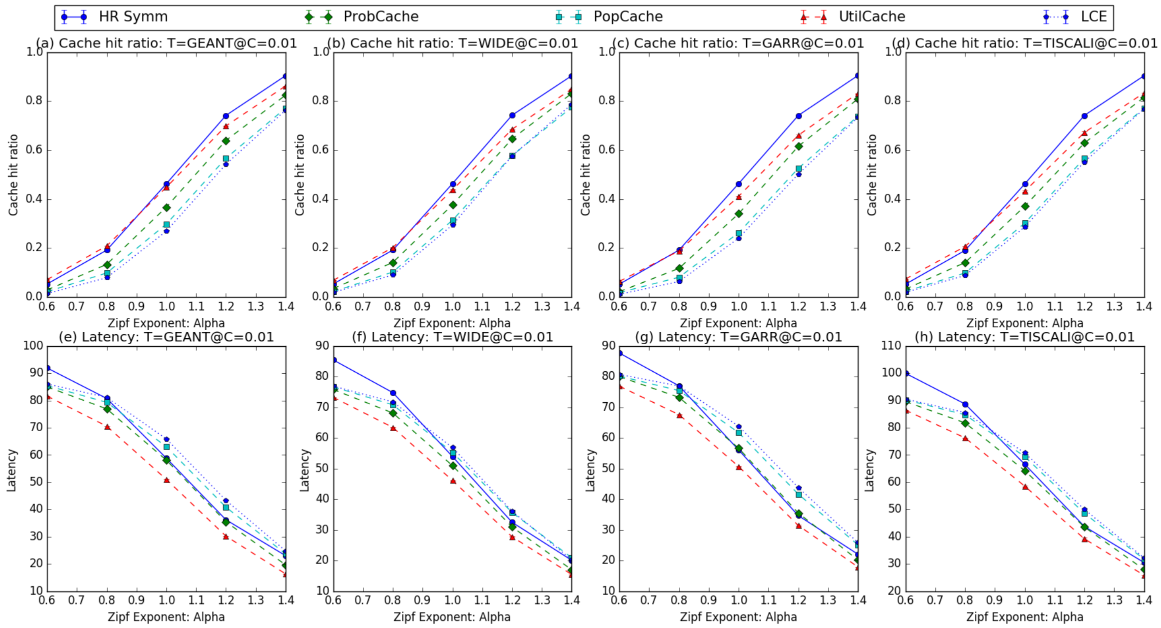


Fig. 4. Cache hit ratio and Latency with $\alpha \in \{0.6, 0.8, 1.0, 1.2, 1.4\}$ and cache to population ratio 1%

We evaluate the performance of five schemes when cache to population ratio $C \in \{0.2\%, 0.4\%, 1\%, 3\%, 5\%\}$ with fixed at $\alpha = 1.0$. The results are similar to those in Fig. 4: we achieve second-best performance in cache hit ratio and best in latency. Due to space constraints, we do not present the figure in our paper. Based on the experiments, we draw a conclusion that UtilCache outperforms other on-path collaborative caching schemes with comparable cache hit ratio and competitive content retrieval latency.

V. Conclusion

In this paper, we formulate the *traffic cost minimization* problem in ICN on-path collaborative caching (TCM-OP). From the formulation we demonstrate that *popularity* of items and *distance* to serving router are two significant factors that need taking into account when dealing TCM-OP problem. Thereby we propose a *caching utility function* concerning popularity and distance to evaluate the payoff of caching an item at a router. Based on the main idea of attaining higher caching utility, we present an on-path collaborative caching scheme UtilCache to address traffic cost minimization problem. UtilCache simply calculates the caching utility for each item at each router, chooses the most utilitarian ones to cache, and evicts the least utilitarian ones. In simulation-based comparison with state of the art, UtilCache yields higher cache hit ratio and lower latency than other on-path caching schemes, which indicates the effectiveness of caching utility function. Nevertheless, UtilCache is merely in fact a greedy mechanism aiming at higher caching utility. Further work is supposed to be done to maximize the caching utility and get content placement approximating closer to optimal solution of formulation TCM-OP.

References

[1] J. Dai, Z. Hu, B. Li, J. Liu, and B. Li, "Collaborative hierarchical caching with dynamic request routing for massive content distribution," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 2444–2452.

[2] M. Dehghan, L. Massoulie, D. Towsley, D. Menasche, and Y. Tay, "A utility optimization approach to network cache design," *arXiv preprint arXiv:1601.06838*, 2016.

[3] "Named data network project," http://www.named-data.net/.

[4] L. Saino, I. Psaras, and G. Pavlou, "Hash-routing schemes for information centric networking," in *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*. ACM, 2013, pp. 27–32.

[5] J. M. Wang, J. Zhang, and B. Bensaou, "Intra-as cooperative caching for content-centric networks," in *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking*. ACM, 2013, pp. 61–66.

[6] T. Mick, R. Tourani, and S. Misra, "Muncc: Multi-hop neighborhood collaborative caching in information centric networks," in *Proceedings of the 3rd ACM Conference on Information-Centric Networking*, 2016, pp. 93–101.

[7] N. Laoutaris, S. Syntila, and I. Stavrakakis, "Meta algorithms for hierarchical web caches," in *Performance, Computing, and Communications, 2004 IEEE International Conference on*. IEEE, 2004, pp. 445–452.

[8] 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*. ACM, 2012, pp. 55–60.

[9] 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*. IEEE, 2013, pp. 236–243.

[10] N. Laoutaris, H. Che, and I. Stavrakakis, "The lcd interconnection of lru caches and its analysis," *Performance Evaluation*, vol. 63, no. 7, pp. 609–634, 2006.

[11] L. Saino, I. Psaras, and G. Pavlou, "Icarus: a caching simulator for information centric networking (icn)," in *Proceedings of the 7th International ICST Conference on Simulation Tools and Techniques*, ser. SIMUTOOLS '14. ICST, Brussels, Belgium, Belgium: ICST, 2014.

[12] "Geant project website," http://www.geant.net/.

[13] "Wide project website," http://www.wide.ad.jp/.

[14] "Garr project website," http://www.garr.it/it/.

[15] "Tiscali: pan-european commercial isp," http://investors.tiscali.it/it/.