Jointly Optimizing Throughput and Content Delivery Cost Over Lossy Cache Networks

Weibo Chu, Zhiwen Yu, *Senior Member, IEEE*, John C. S. Lui, *Fellow, IEEE*, and Yi Lin

***Abstract*— Cache optimization, i.e., determining the optimal content placement and routing paths, is essential for obtaining high performance of cache-enabled networks. This paper studies the problem of optimizing system throughput and content delivery cost over cache networks with lossy links (i.e., ICN-based wireless IoT systems), where content is divided into packet-level chunks, and packets may be lost in transmission. We first propose a new performance metric – *the expected overall content routing cost for satisfied requests* (RCS), for better characterizing content delivery cost under packet losses. RCS at the same time possesses the attractive mathematical property of super-modularity. We then formulate an optimization problem for the task through jointly optimizing content caching and request routing, and analyze it under fixed-routing scenario. The formulated problem is NP-hard and we prove it is reducible to the one of minimizing content routing cost without packet losses. We establish rules for the reduction, and leverage existing efficient algorithm to solve the problem. We also propose a *potential*-based online algorithm that is simple and adaptive to traffic changes and packet losses. The effectiveness of our mechanism is validated through extensive simulations over a wide array of network topologies.**

***Index Terms*— Cache optimization, throughput, routing cost, adaptive algorithm.**

I. Introduction

Driven by the ever-increasing network traffic [3] and the promotion/advancement of new technologies such as 5G [43] and edge computing [39], [42], cache-enabled networks have attracted a lot of attention in recent years. Micro/femtocell networks, content delivery networks (CDNs), peer-to-peer networks and Information-Centric Networks (ICNs) [5], [26], are some examples of cache-enabled networks. A common feature of these networks is that in addition to routing capability, network nodes are augmented with additional storage resources so that they can provide data and act as content sources as well. This brings significant performance benefits such as shorter delays, less congestion and better mobility.

A fundamental research problem in this area is how to place content items in cache nodes and how to route requests towards them in order to maximize the network performance. This problem, sometimes referred to as cache optimization [37], has attracted a lot of recent studies. Examples include modeling and characterizing caching dynamics [36], [46], design and performance evaluation of caching mechanisms [10], [19], to name a few. Among them, some treat content placement and request routing separately, whereas others consider them jointly.

A major issue with the prior work on cache optimization is that they generally assumed perfect links (content delivery) between nodes and ignored packet losses, i.e., a request for content will always be routed to content sources and the corresponding data will always be delivered back to the user/client, which is not a realistic setting (see e.g., [24], [18]). In fact, it is constantly reported that network failures and packet losses are common events in real networks such as the Internet [35]. This implies overestimated performance of existing mechanisms if they were developed without considering these factors.

Another issue is that prior work generally focused on optimizing network-side performance metrics such as overall traffic routing cost, total network energy consumption. While these are critical performance metrics of interest, we argue that in some applications such as IoT systems [22], [30], mobile computing [40], device-to-device (D2D) communications [27], [28], client-side metrics are equally or even more important due to the limited resources at the client-end. For example, in a cache-enabled C-RAN architecture for IoT sensing services [45], caching content at the edge not only alleviates the network traffic, but also improves client-side performance as it avoids activating sensors too frequently, which results in a low energy consumption for transmitting sensed data. Obviously, for these systems optimizing client-side performance is far more important as system lifetime is usually determined by client energy consumptions.

Motivated by the above limitations, in this paper we study the problem of optimizing content delivery over a *lossy* cache network, taking both user/client and network operator's interests into account. We consider a setting where content is divided into packet-level chunks (e.g., NDN-like data [2] [47]), and packets (request or data) may be lost in transmission

Manuscript received May 6, 2020; revised October 18, 2020 and January 2, 2021; accepted February 16, 2021. Date of publication February 24, 2021; date of current version June 16, 2021. This work was supported in part by the National Science Fund for Distinguished Young Scholars (61725205), the National Key R&D Program of China (2019YFB2102200), and the National Natural Science Foundation of China (61960206008). The work of John C. S. Lui was supported in part by the GRF 14200117. The associate editor coordinating the review of this article and approving it for publication was T. He. *(Corresponding author: Weibo Chu.)*

Weibo Chu, Zhiwen Yu, and Yi Lin are with the School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: wbchu@nwpu.edu.cn; zhiwenyu@nwpu.edu.cn; ly_cs@nwpu.edu.cn).

John C. S. Lui is with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong (e-mail: cslui@cse.cuhk.edu.hk).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2021.3061685.

Digital Object Identifier 10.1109/TCOMM.2021.3061685

0090-6778 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

between nodes due to congestion, node mobility, media errors. We take system throughput as a key performance metric to capture client-side interest, for the fact that a high throughput usually implies less packet transmissions and low energy consumption. Note that an interesting problem lies here as it remains unclear how to maximize throughput in a cache network. This paper provides theoretical analysis and an algorithmic solution to this interesting problem.

Furthermore, we take overall routing cost to capture network operator's interest. Our simple analysis shows that optimizing this performance metric in a lossy environment is challenging, since it brings two new problems: 1) it would drive the network to forward requests over links with high packet loss rate, which is not acceptable in real network, and 2) it does not necessarily guarantee attractive mathematical properties that facilitate us to design efficient solution mechanisms.

In this paper, we study the problem of optimizing throughput and content routing cost over a lossy cache network such as ICN-based wireless IoT system [7]. We propose a new performance metric that tackles the problem we observed in our analysis, and formulate an optimization problem with the two performance metrics taken into account, through jointly optimizing content caching and request routing. We theoretically analyze the problem under *fixed-routing scenario*, and propose online algorithmic solutions to the problem.

More specifically, we make the following contributions:

1) We analyze the metric of content routing cost in a lossy cache network, and find that directly optimizing this performance metric without explicitly considering packet losses in network will result in a low throughput as it would drive the network to forward content items along paths with high loss rate, which is not acceptable.

2) We propose a new performance metric – *the expected overall content routing cost for satisfied requests* (RCS), that reflects both the heterogeneous routing cost and packet loss rates over links. We prove that RCS possesses the attractive mathematical property of supermodularity. This allows us to obtain solutions with provable performance guarantees by leveraging existing efficient algorithms, i.e., within $(1 - 1/e)$ from the optimum [4].

3) We formulate a problem for maximizing a weighted sum of throughput and routing cost gain (using RCS) in a lossy cache network, through jointly optimizing request routing and content caching. We prove that the problem is NP-hard in general, and analyze it under fixed-routing scenario, i.e., when the path to deliver each request/item is fixed. We then prove this problem can be reduced to the original problem of minimizing content routing cost alone in a network without packet losses, by adopting the concept of "*virtual weight*". We present rules for the reduction, and further demonstrate how the existing projected sub-gradient ascend algorithm can be adapted to solve our problem with provable performance guarantees.

4) We observe that the existing distributed online algorithm has some drawbacks, i.e., additional traffic routing cost is incurred in cache reshuffling, and then propose a new *potential*-based online algorithm that is simple and adaptive to changes and packet losses. This algorithm is fairly general as it allows us to optimize other performance metrics in content delivery as well, through properly defining potentials.

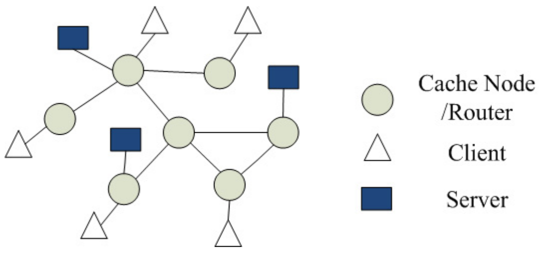


Fig. 1. A general cache network.

5) We evaluate our proposed algorithm as well as the adapted sub-gradient based algorithm through extensive simulations. Results indicate that they are effective under a wide array of network topologies and system settings, with different cache replacement policies and path replication strategies. For example, as compared to traditional caching algorithms such as LRU, our potential-based algorithm improves system throughput by up to 18%, and reduces content routing cost by as much as 50%, under the single-path routing scenario with severe packet losses.

The remainder of this paper is organized as follows. In Section II we introduce our model and present analysis and insights of the two performance metrics. We then formulate the joint throughput and content delivery cost optimization problem. In Section III we prove fundamental properties of the formulated problem under the fixed-path routing scenario, give rules for problem reduction and also elaborate the algorithmic solutions. In Section IV we present numerical results via simulations. Section V reviews related work and we conclude the paper in Section VI.

II. Model and Problem Description

A. Model

We consider a cache network represented by a graph $G(V, E)$ as shown in Fig 1, where V denotes the set of nodes and E is the set of edges between nodes.[1] Each node $v \in V$ is equipped with some cache resource of size C_v. The set of content items distributed over the network is denoted by $\mathcal{C}$.

The network serves user requests by forwarding them to content servers. For each content item $i \in \mathcal{C}$, let $\mathcal{S}_i$ be the set of nodes (content servers) that permanently store i. The set of all content servers is denoted as $\mathcal{S} = \bigcup_{i \in \mathcal{C}} \mathcal{S}_i$.

A request is forwarded until either it reaches a content server, or an intermediate node with the requested item in cache. Once this happens, the requested item is then sent back to the user along the same path in reverse direction. We characterize each request by a tuple (i, s), where $i \in \mathcal{C}$ is the item requested, and $s \in V$ is the source node where the request originates. The set of all requests is denoted by $\mathcal{R} \subseteq \mathcal{C} \times V$. We assume requests for item i at source node s arrive independently and that they follow a Poisson process with rate $\lambda_{(i,s)}$.

[1]We use node, router and cache interchangeably throughout this paper.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

In this work, we consider content items are of *equal packet-level sizes* (i.e., as NDN-like [2] [47] data), which implies that each node v can hold at most C_v items. We associate each node $v \in V$ with a vector $X_v = [x_{vi}] \in \{0,1\}^{|\mathcal{C}|}$, where x_{vi} denotes whether node v caches item i. These variables satisfy the following cache capacity constraint:

$$\sum_{i\in\mathcal{C}} x_{vi} \leq C_v, \quad \text{for all } v \in V \setminus \mathcal{S}. \tag{1}$$

Note that for each request (i,s), there are potentially multiple paths that direct it to content servers. Let $\mathcal{P}_{(i,s)}$ be the set of routing paths for (i,s). Each path $p \in \mathcal{P}_{(i,s)}$ is comprised of a sequence of nodes $(p_1, p_2, \ldots, p_K)$ of length $|p| = K-1$, where $p_1 = s$, $p_K \in \mathcal{S}_i$, and $(p_k, p_{k+1}) \in E$ for $k = 1, 2, \ldots, K-1$. We further assume that (i,s) is *well-routed*, that is: 1) p is loop-free, and 2) $p_k \notin \mathcal{S}_i$ for $k = 1, 2, \ldots, K-1$, i.e., only the last node in p is a content server for i.

Let $r_{(i,s)} = [r_{(i,s),p}] \in \{0,1\}^{|\mathcal{P}_{(i,s)}|}$ be a routing vector for each request (i,s), where $r_{(i,s),p}$ denotes whether path $p \in \mathcal{P}_{(i,s)}$ is selected to forward (i,s), i.e., $r_{(i,s),p} = 1$ when p is selected. To ensure that every user request is forwarded, these routing variables should satisfy:

$$\sum_{p\in\mathcal{P}_{(i,s)}} r_{(i,s),p} = 1, \quad \text{for all } (i,s) \in \mathcal{R}. \tag{2}$$

To make our model more realistic, we consider networks with *lossy links*, where packets (either request or data) may be lost in transmission. While a number of factors (e.g., congestion, media errors, or node mobility) can cause packet losses and precisely predicting packet loss event is out of the scope of this work, we assume packet loss events over a link and across links are independent, and simply adopt $f_{ij} \in [0,1]$ – the packet loss rate over link $(i,j) \in E$, to characterize the probability that a packet will not be successfully delivered over (i,j) in transmission.

A consequence of lossy links is that requests for items may not be satisfied[2] and users will then have to re-issue them to fetch the content, which incurs extra network cost. From users' point of view, more requests for the same content item also implies higher energy consumption and longer delays. As a result, the rate of requests satisfied by network given user demand, also referred to as *system throughput*, becomes a key performance metric to optimize. Note that as compared to traditional networks, a cache network should provide more opportunities for requests to be satisfied and hence more throughput. But questions such as *what is its performance under a lossy environment, how much improvement a lossy cache network can provide and how to maximize the throughput*, remain unsolved.

In addition to system throughput, we also consider network-side metric, i.e., *content routing cost*. Optimizing this performance metric in a cache network has been well studied (see, e.g., [24], [25], [34]), and both centralized and distributed solutions have been proposed. However, we observe that in a lossy environment the problem becomes very different and challenging to solve. In particular, the very property of *sub-modularity* of the objective function in formulated optimization models, which is the key to derive efficient solution mechanisms [25], [41], no longer exists. Furthermore, as we show in the later section, directly adopting existing performance metric would drive the network to forward requests over those links with high packet loss rates, which is obviously unacceptable in practice.

In this paper, we seek to answer the aforementioned questions by formally addressing the problem of optimizing content delivery over a cache network with lossy links. Specifically, we propose a new network-side performance metric, which not only can reflect the heterogeneous content routing cost and packet loss rates over links, but also possess attractive mathematical properties that facilitates us to design efficient solution mechanisms. We take the *system throughput* and *content routing cost for satisfied requests* as two key metrics for the design and optimization. Our goal is to maximize system throughput over a lossy cache network while at the same time, keep the overall content routing cost for the satisfied requests as low as possible, by jointly optimizing content caching and request routing. We consider both centralized and distributed/adaptive solutions with optimality guarantees.

TABLE I
MAIN NOTATIONS

$G(V,E)$	Network graph with node set V and edge set E
$\mathcal{C}$	Set of content items
C_v	Cache capacity at node v
$\mathcal{S}_i$	Set of content servers for item $i \in \mathcal{C}$
$\mathcal{S}$	Set of all content servers
(i,s)	A request for content item i that originates at node s
$\lambda_{(i,s)}$	Request rate for content item i at node s
$\mathcal{R}$	Set of all requests
$\mathcal{P}_{(i,s)}$	Set of routing paths for request (i,s)
w_{uv}	Weight of edge $(u,v) \in E$
f_{uv}	Packet loss rate over link $(u,v) \in E$
x_{vi}	Caching var. denoting whether node v caches item i
$r_{(i,s),p}$	Routing var. denoting whether (i,s) is routed over path p
$TP(r,X)$	Throughput under routing strategy r & caching strategy X
$RCS(r,X)$	The expected overall cost for satisfied reqs under r and X

B. Problem Formulation

Let $X = [x_{vi}]_{v\in V,\, i\in\mathcal{C}}$ and $r = [r_{(i,s),p}]_{(i,s)\in\mathcal{R}, p\in\mathcal{P}_{(i,s)}}$ be the *global* caching strategy and routing strategy, respectively. To calculate system throughput, we first focus on an arbitrary request $(i,s) \in \mathcal{R}$ routed over a path $p \in \mathcal{P}_{(i,s)}$, as shown in Fig. 2. Since p is well-routed, a node p_k in p can serve (i,s) if and only if the following conditions hold: 1) the request (i,s) is successively delivered to node p_k; 2) all cache nodes preceding p_k, i.e., p_1, p_2, …, p_{k-1}, do not hold item i, and 3) node p_k has i in its cache. The probability that (i,s) is delivered to p_k is $\prod_{l=1}^{k-1}(1 - f_{p_l p_{l+1}})$. Once node p_k serves (i,s), a data packet is generated. The probability that this data packet will be delivered back to s is $\prod_{l=1}^{k-1}(1 - f_{p_{l+1} p_l})$. As a result, the probability that (i,s) is satisfied by p_k can be

[2]By being satisfied we mean that a request is responded by some node and its corresponding data is received at the source node.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

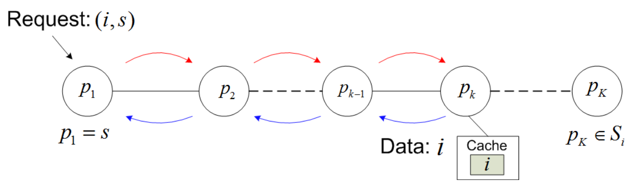


Fig. 2. A request (i, s) is forwarded over path $p = (p_1, p_2, \ldots, p_K)$ and responded by node p_k.

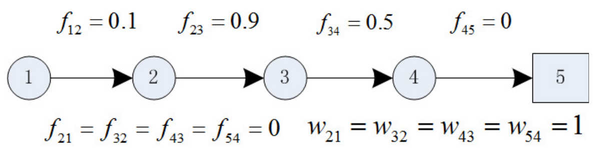

$f_{12} = 0.1$
$f_{23} = 0.9$
$f_{34} = 0.5$
$f_{45} = 0$
1
2
3
4
5
$f_{21} = f_{32} = f_{43} = f_{54} = 0$
$w_{21} = w_{32} = w_{43} = w_{54} = 1$

Fig. 3. A path network. Source node 1 generates request for one item to end server 5. Node 2, 3, and 4 are cache nodes that can only accommodate 1 item.

expressed as follows:

$$PR_{(i,s),p,k}(X) = \prod_{l=1}^{k-1}(1 - f_{p_l p_{l+1}})(1 - f_{p_{l+1} p_l}) \times (1 - x_{p_l i})x_{p_k i}. \tag{3}$$

The probability for (i, s) being satisfied over path p can be given by summing over all nodes in p:

$$PR_{(i,s),p}(X) = \sum_{k=1}^{|p|} PR_{(i,s),p,k}(X), \tag{4}$$

and the probability for (i, s) being satisfied by network is therefore:

$$PR_{(i,s)}(r, X) = \sum_{p\in\mathcal{P}_{(i,s)}} r_{(i,s),p} \sum_{k=1}^{|p|} PR_{(i,s),p,k}(X). \tag{5}$$

System throughput $TP(r, X)$ then can be derived as:

$$TP(r, X) = \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \times PR_{(i,s)}(r, X). \tag{6}$$

Let

$$A_{(p,k)} = \prod_{l=1}^{k-1}(1 - f_{p_l p_{l+1}})(1 - f_{p_{l+1} p_l}) \tag{7}$$

be the probability that a request responded by node p_k in path p gets satisfied, and I_0 be a constant defined below:

$$I_0 = \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \sum_{p\in\mathcal{P}_{(i,s)}} (1 - A_{(p,|p|)}), \tag{8}$$

then

$$\begin{aligned} I_0 - \left[\sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} - TP(r, X)\right] &= \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \left[\sum_{p\in\mathcal{P}_{(i,s)}} (1 - A_{(p,|p|)}) \right. \\ &\quad \left. - (1 - \sum_{p\in\mathcal{P}_{(i,s)}} r_{(i,s),p} \times PR_{(i,s),p}(X))\right]. \end{aligned} \tag{9}$$

It is not hard to see that I_0 is an upper bound of $\sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} - TP(r, X)$, and maximizing $TP(r, X)$ is equivalent to maximizing $I_0 - (\sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} - TP(r, X))$. For this reason, we define $I_0 - (\sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} - TP(r, X))$ as the ***throughput gain*** due to in-network caching and path selection.

Next, we consider content routing cost. Following the common practice, we ignore the routing cost for each request and only consider that for response/data packets. Let $w_{ij} > 0$ be the weight of link (i, j), which denotes the cost (e.g., delay, energy) of delivering a data packet over (i, j). Once the data to request (i, s) is generated by node p_k, the expected routing cost for delivering it back to the source node over path p can be calculated as:

$$RC_{(i,s),p,k}(X) = \left(\sum_{m=2}^{k} w_{p_m p_{m-1}} \prod_{n=m+1}^{k} (1 - f_{p_n p_{n-1}})\right) \left(\prod_{l=1}^{k-1}(1 - x_{p_l i})x_{p_k i}\right), \tag{10}$$

Incorporating the probability that (i, s) is delivered to p_k, we have the expected routing cost incurred by (i, s) over p as:

$$RC_{(i,s),p}(X) = \sum_{k=1}^{|p|-1} \left[RC_{(i,s),p,k}(X) \times \prod_{l=1}^{k-1}(1 - f_{p_l p_{l+1}})\right], \tag{11}$$

We want to note that the above definition of routing cost brings significant challenges for us to optimize as it does not necessarily guarantee good mathematical property which helps devise efficient solution mechanisms. More specifically, depending on the values of packet loss rates and weights over links, the attractive property of sub-modularity or super-modularity, which is the basis of existing solution mechanisms, may not exist.

To illustrate, consider the example in Fig. 3 where node 1 is the source node generating requests for an item hosted at node 5 (content server). Node 2, 3 and 4 are cache nodes that can hold only one item. Let $X = \{0, 1\}^{1\times 3}$ be the item placement over the 3 cache nodes. We have $RC_{(i,s),p}([0, 0, 0]) = (1 - f_{12})(1 - f_{23})(1 - f_{34})(w_{54} + w_{43} + w_{32} + w_{21}) = 0.18$, $RC_{(i,s),p}([1, 1, 0]) = RC_{(i,s),p}([1, 0, 0]) = (1 - f_{12}) \times w_{21} = 0.9$, $RC_{(i,s),p}([0, 1, 1]) = RC_{(i,s),p}([0, 1, 0]) = (1 - f_{12})(1 - f_{23})(w_{32} + w_{21}) = 0.18$, $RC_{(i,s),p}([0, 0, 1]) = (1 - f_{12})(1 - f_{23})(1 - f_{34})(w_{43} + w_{32} + w_{21}) = 0.135$.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

Note that $RC_{(i,s),p}(X)$ can also be expressed as a set function $RC_{(i,s),p}(U_X)$ with the support of the binary vector X being the set U_X, i.e., $RC_{(i,s),p}(\{2,3\}) = RC_{(i,s),p}([0,1,1])$. The above example indicates that $RC_{(i,s),p}(U_X)$ is neither a sub-modular set function nor a super-modular set function since $RC_{(i,s),p}(\{2,4\}) - RC_{(i,s),p}(\{4\}) > RC_{(i,s),p}(\{2,3,4\}) - RC_{(i,s),p}(\{3,4\})$, $RC_{(i,s),p}(\{2\}) - RC_{(i,s),p}(\phi) < RC_{(i,s),p}(\{2,4\}) - RC_{(i,s),p}(\{4\})$.

Another problem when we use $RC_{(i,s),p}(X)$ is that it would drive the network to forward requests over links with high packet loss rates. For the above example, the optimal solution is that node 4 caches the item. This solution, however, implies that only 4.5% requests can be satisfied, whereas caching the item at node 2 would satisfy 90%.

To overcome the above limitations, in this paper, we propose a new network-side performance metric, ***the expected overall routing cost for satisfied requests*** (RCS for short hereafter), that captures both the content routing cost and user requests satisfaction. Assume that users will re-issue requests whenever a request or a data packet is lost in transmission. Then the quantity $\sum_{m=1}^{k-1} \frac{w_{p_{k-m+1}p_{k-m}}}{\prod_{n=1}^{k-m-1}(1-f_{p_{n+1}p_n})}$, which characterizes content routing cost when (i,s) is responded by node p_k, corresponds to the expected content routing cost incurred for (i,s) until the request gets satisfied. We define RCS as the expected overall content routing cost when all user demands are satisfied, which can be expressed as:

$$RCS(r,X) = \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \sum_{p\in\mathcal{P}_{(i,s)}} r_{(i,s),p} \sum_{k=1}^{|p|} \times \left(\sum_{m=1}^{k-1} \frac{w_{p_{k-m+1}p_{k-m}}}{\prod_{n=1}^{k-m-1}(1-f_{p_{n+1}p_n})}\right) \times \left(\prod_{l=1}^{k-1}(1-x_{p_l i})x_{p_k i}\right). \tag{12}$$

The mathematical derivation of RCS is as follows. Let us again consider the request (i,s) routed over path p and responded by node p_k. Recall that the expected routing cost for delivering the data back to the source node is $\sum_{m=2}^{k} w_{p_m p_{m-1}} \prod_{n=m+1}^{k}(1-f_{p_n p_{n-1}})$. We divide it by the probability that this request gets satisfied when the response is generated by node p_k, that is $\prod_{n=1}^{k-1}(1-f_{p_{n+1}p_n})$, and denote this quantity by $RCS_{(i,s),p,k}(X)$:

$$\begin{aligned} RCS_{(i,s),p,k}(X) &= \frac{\sum_{m=2}^{k} w_{p_m p_{m-1}} \prod_{n=m+1}^{k}(1-f_{p_n p_{n-1}})}{\prod_{n=1}^{k-1}(1-f_{p_{n+1}p_n})} \times \left(\prod_{l=1}^{k-1}(1-x_{p_l i})x_{p_k i}\right) \\ &= \left(\sum_{m=1}^{k-1} \frac{w_{p_{k-m+1}p_{k-m}}}{\prod_{n=1}^{k-m-1}(1-f_{p_{n+1}p_n})}\right) \times \left(\prod_{l=1}^{k-1}(1-x_{p_l i})x_{p_k i}\right). \end{aligned} \tag{13}$$

RCS for (i,s) being satisfied by path p is defined as:

$$RCS_{(i,s),p}(X) = \sum_{k=1}^{|p|} RCS_{(i,s),p,k}(X), \tag{14}$$

and that for (i,s) being satisfied by network is then:

$$RCS_{(i,s)}(r,X) = \sum_{p\in\mathcal{P}_{(i,s)}} r_{(i,s),p} \times RCS_{(i,s),p}(X), \tag{15}$$

RCS for all requests in network is therefore:

$$RCS(r,X) = \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \times RCS_{(i,s)}(r,X), \tag{16}$$

which turns out to be Eq. (12).

Lemma 1: $RCS(r,X)$ is supermodular.

Proof: Since $RCS(r,X)$ is defined as the summation over all request packets and all delivery paths, it suffices to prove that RCS is supermodular for each request that delivered over a path. Let S_1 and S_2 be the set of nodes in path p holding item i, $S_1 \subset S_2$, and N_{S_1} (N_{S_2}) be the node in S_1 (S_2) which is closest to the source node s. We denote $N_{S_2} \preceq N_{S_1}$ to refer to the fact that N_{S_2} is not farther away than N_{S_1} from the source node. Due to the caching effect, we have $RCS_{(i,s),p,k}(S_1) = RCS_{(i,s),p,k}(\{N_{S_1}\})$, $RCS_{(i,s),p,k}(S_2) = RCS_{(i,s),p,k}(\{N_{S_2}\})$. Meanwhile, since RCS is monotone (see Eq. (13)) in terms of caching positions of the delivery path, $N_{S_2} \preceq N_{S_1} \Rightarrow RCS_{(i,s),p,k}(S_2) \le RCS_{(i,s),p,k}(S_1)$.

Let v be a node in path p and $v \notin S_2$. If $N_{\{v\}} \preceq N_{S_2} \preceq N_{S_1}$, then $N_{S_1\cup\{v\}} = N_{S_2\cup\{v\}}$, and $RCS_{(i,s),p,k}(S_1\cup\{v\}) = RCS_{(i,s),p,k}(S_2\cup\{v\})$, then we have $RCS_{(i,s),p,k}(S_1\cup\{v\}) - RCS_{(i,s),p,k}(S_1) \le RCS_{(i,s),p,k}(S_2\cup\{v\}) - RCS_{(i,s),p,k}(S_2)$. If $N_{S_2} \preceq N_{\{v\}} \preceq N_{S_1}$, then $RCS_{(i,s),p,k}(S_1\cup\{v\}) - RCS_{(i,s),p,k}(S_1) \le 0 = RCS_{(i,s),p,k}(S_2\cup\{v\}) - RCS_{(i,s),p,k}(S_2)$. If $N_{S_2} \preceq N_{S_1} \preceq N_{\{v\}}$, then $RCS_{(i,s),p,k}(S_1\cup\{v\}) - RCS_{(i,s),p,k}(S_1) = RCS_{(i,s),p,k}(S_2\cup\{v\}) - RCS_{(i,s),p,k}(S_2) = 0$. Summarizing all the above three cases, we have $RCS_{(i,s),p,k}(S_1\cup\{v\}) - RCS_{(i,s),p,k}(S_1) \le RCS_{(i,s),p,k}(S_2\cup\{v\}) - RCS_{(i,s),p,k}(S_2)$, which implies supermodularity. □

Let $C_{(i,s),p}$ be the expected content routing cost when request (i,s) is responded by the content server at the end of path p, i.e.,

$$C_{(i,s),p} = \sum_{m=1}^{|p|-1} \frac{w_{p_{|p|-m+1}p_{|p|-m}}}{\prod_{n=1}^{|p|-m-1}(1-f_{p_{n+1}p_n})}, \tag{17}$$

and C_0 be a constant given as follows:

$$C_0 = \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \sum_{p\in\mathcal{P}_{(i,s)}} C_{(i,s),p}. \tag{18}$$

Similarly, we can see that C_0 is an upper bound of $RCS(r,X)$, and $C_0 - RCS(r,X)$ is the ***routing cost gain*** due to in-network caching and path selection.

Lemma 2: Minimizing RCS in a lossy cache network can be sub-optimal when the goal is to maximize throughput.

Proof: We prove the lemma by giving a simple example as shown in Fig. 4. The example is about two source nodes generating requests at the same rate to two end-servers through

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

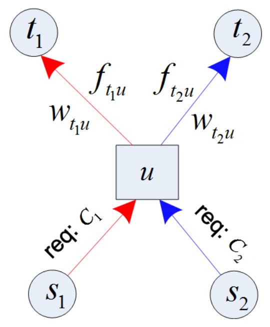


Fig. 4. A simple cache network. Source node s_1 (s_2) generates request for one item to end servers t_1 (t_2). Intermediate node u can only accommodate 1 item. $w_{t_1u} = 1, w_{t_2u} = 5,\ f_{t_1u} = 0.3, f_{t_2u} = 0.1$.

a common cache node u which is able to accommodate 1 content item. Source node u_1 (u_2) continually issues requests to item c_1 (c_2) hosted at end-server t_1 (t_2). Denote by w_{t_1u} (w_{t_2u}) the content routing cost of delivering an item over link (t_1, u) ((t_2, u)), and f_{t_1u} (f_{t_2u}) the corresponding packet loss rate over links. Set $w_{t_1u} = 1, w_{t_2u} = 5,\ f_{t_1u} = 0.3, f_{t_2u} = 0.1$, and assume content routing cost and packet loss rates over other links are identical. It can be easily verified that to minimize RCS we should have node u cache the item c_2. However, c_1 should be stored at u if our goal is to optimize throughput. □

The above example shows that optimizing throughput and minimizing RCS are two different goals, and sometimes they can even be in conflict. A natural question then arises as how to optimize content delivery over a lossy cache network with both metrics taken into consideration, i.e., maximizing system throughput while at the same time, keeping the overall routing cost as low as possible, and balance them whenever required. Formally, we can formulate the MAX-TP-RCS as the following optimization problem with the objective function being the weighted sum of the throughput gain ($I_0 - \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} + TP(r, X)$) and routing cost gain ($C_0 - RCS(r, X)$), which is:

$$\begin{aligned}\text{Max:}\quad & \alpha \times [I_0 - \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} + TP(r, X)] \\ & + (1-\alpha) \times [C_0 - RCS(r, X)] \qquad (19a)\\ \text{subj. to:}\quad & (r, X) \in \mathcal{D} \qquad (19b)\end{aligned}$$

where $\mathcal{D}$ is the set of (r, X) satisfying the following routing, capacity and integrality constraints:

$$\sum_{i\in\mathcal{C}} x_{vi} \le C_v, \quad \text{for all } v \in V \setminus \mathcal{S}, \qquad (20a)$$

$$\sum_{p\in\mathcal{P}_{(i,s)}} r_{(i,s),p} = 1, \quad \text{for all } (i, s) \in \mathcal{R}, \qquad (20b)$$

$$x_{vi} \in \{0, 1\}, \quad \text{for all } i \in \mathcal{C} \text{ and } v \in V, \qquad (20c)$$

$$x_{vi} = 1, \quad \text{for all } i \in \mathcal{C} \text{ and } v \in \mathcal{S}_i, \qquad (20d)$$

$$r_{(i,s),p} \in \{0, 1\}, \quad \text{for all } (i, s) \in \mathcal{R} \text{ and } p \in \mathcal{P}_{(i,s)}. \qquad (20e)$$

and $\alpha \in [0, 1]$ is a constant denoting the relative weight between the throughput gain and the routing cost gain. When $\alpha = 0$, the problem becomes that of minimizing the expected overall routing cost, while $\alpha = 1$ means the goal is to maximize throughput.

Problem (19) is combinatorial in nature and is NP-hard as we can prove it later. In a network with a large set of nodes and content items, it is prohibitively costly to obtain exact solutions. Furthermore, centralized mechanisms only work when all parameters are given a prior, which may not be possible in real networks (i.e., network topology may change and demand variation frequently occurs). Therefore, we seek distributed algorithms that adapt to these changes and at the same time, provide theoretical performance guarantees.

III. Solution

In this section, we analyze problem MAX-TP-RCS under fixed-routing scenario, i.e., when request routing is both fixed and deterministic, and derive its algorithmic solutions.

A. Analysis

Without loss of generality, under fixed-routing scenario we assume each request (i, s) is forwarded solely over a path $p = p_{(i,s)}$. The problem MAX-TP-RCS then reduces to MAX-TP-RCS-FR[3] as follows:

$$\begin{aligned}\text{Max: } & \alpha \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} [PR_{(i,s),p}(X) - A_{(p,|p|)}] \\ & + (1-\alpha) \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \left(C_{(i,s),p} - RCS_{(i,s),p}(X)\right) \qquad (21a)\\ \text{subj. to: } & (20a),\ (20c),\ (20d) \qquad (21b)\end{aligned}$$

where $A_{(p,|p|)}$ is given by Eq. (7).

Theorem 1: Problem MAX-TP-RCS-FR is equivalent to Problem: $\max F(X)$ with the same constraints (20a), (20c), (20d), where:

$$F(X) = \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \sum_{k=1}^{|p|-1} B_{(p,k)} (1 - \prod_{l=1}^{k} (1 - x_{p_l i})) \qquad (22)$$

and

$$B_{(p,k)} = \alpha \times (A_{(p,k)} - A_{(p,k+1)}) + \frac{(1-\alpha) \times w_{p_{k+1}p_k}}{\prod_{l=1}^{k-1} (1 - f_{p_{l+1}p_l})} \qquad (23)$$

Proof: Let $\mathcal{D}'$ be the feasible solution set given by constraints (20a), (20c), (20d), and $G(X)$ be the objective function of Problem MAX-TP-RCS-FR. Since the two problems have the same constraints, the theorem holds if $G(X) = F(X), \forall X \in \mathcal{D}'$. We next show that this is indeed the case. Denote by $G_{(i,s),k}(X)$ and $F_{(i,s),k}(X)$ be the contribution to $F(X)$ and $G(X)$ respectively when (i, s) is served by node p_k in path p, i.e., when $x_{p_1 i} = \ldots = x_{p_{k-1} i} = 0$, $x_{p_k i} = 1$. Then it can be easily verified that $G_{(i,s),k}(X) = F_{(i,s),k}(X)$,

[3]Here FR stands for fixed routing.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

$k = 1, 2, \ldots, |p|$. The proof completes since both $G(X)$ and $F(X)$ are separable in terms of requests. □

With the new objective function $F(X)$, we now compare MAX-TP-RCS-FR to existing optimization problems in cache networks that deal with optimizing some performance metrics with demand distributed across the network, i.e., the one that minimizes the overall routing cost. Recall that the task of minimizing routing cost in a cache network (but with no packet losses) given fixed routing paths can be cast as the following problem of maximizing the caching gain (MAX-CG) [24]:

$$\text{Max:} \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \sum_{k=1}^{|p|-1} w'_{p_{k+1}p_k}(1 - \prod_{k'=1}^{k}(1-x_{p_{k'}i})) \quad (24a)$$
$$\text{subj. to:} \quad (20a), (20c), (20d) \quad (24b)$$

where $w'_{p_{k+1}p_k}$ is the weight[4] of link $(p_{k+1}, p_k) \in E$.

The following theorem shows that by allowing weights to be set w.r.t each request and as a function of packet loss rates and actual weights over links, there exists a *polynomial-time reduction* from Problem MAX-TP-RCS-FR to MAX-CG.

Theorem 2: Problem MAX-TP-RCS-FR and MAX-CG can be transformed to each other at polynomial time complexity.

Proof: We prove by first showing that through properly assigning weights to edges w.r.t each request in problem MAX-CG, we can transform MAX-TP-RCS-FR to MAX-CG. Note that the objective functions of the two problems have the same form. Furthermore, constraints of the two problems are also the same. As a result, transformation from MAX-TP-RCS-FR to MAX-CG, if possible, can be derived by enforcing the following equations for each request $(i, s) \in \mathcal{R}$:

$$w'^{(i,s)}_{p_{k+1}p_k} = B_{(p,k)}, k = 1, 2, \cdots, |p| - 1 \quad (25)$$

Since $B_{(p,k)} > 0$, we can see that for each path p, there exists $\{w'^{(i,s)}_{p_{k+1}k_l}\} > 0$ that can be efficiently computed at time complexity $O(|p|)$. As a result, MAX-TP-RCS-FR can be transformed to MAX-CG, and it takes $O(|\mathcal{R}| \times |p|)$ for transformation. Note that the weights are w.r.t each request while they are the same for all requests in the original MAX-CG problem.

Next consider the reverse direction, that is, to transform MAX-CG to MAX-TP-RCS-FR. We look at the following equation:

$$B_{(p,k)} = w'_{p_{k+1}p_k}, k = 1, 2, \cdots, |p| - 1 \quad (26)$$

It is clear that this equation always hold by setting in $B_{(p,k)}$ $\alpha = 0$, $f_{p_{k+1}p_k} = 0$ and $w_{p_{k+1}p_k} = w'_{p_{k+1}p_k}$ for each $(p_{k+1}, p_k) \in E$. As a result, MAX-CG can also be transformed to MAX-TP-RCS-FR, and it takes time $O(|E|)$ for transformation. The proof completes. □

*Remark: The proof of Theorem 2 is in fact based on the concept of **"virtual weight"**. Together with Theorem 1 we can always define and obtain such non-negative virtual weights as long as the per-path metric (i.e., RCS, throughput) is defined in a way that it is monotone along the path. The non-negativeness of the virtual weight also helps approximate the optimization problem and further develop distributed/adaptive algorithms as we show later.*

[4]We use w' and w to denote weight of edges in problem MAX-CG and MAX-TP-RCS, respectively.

Based on Theorem (2), we immediately have the following results:

Theorem 3: Problem MAX-TP-RCS-FR is NP-hard, and so is MAX-TP-RCS.

Proof: This is due to the fact that MAX-CG is NP-hard [24], and MAX-TP-RCS-FR is a special case of MAX-TP-RCS. □

Theorem 4: There exists approximate algorithm with factor $(1 - 1/e)$ to MAX-TP-RCS-FR.

Proof: This is because MAX-CG can be solved in polynomial time within $(1 - 1/e)$ approximation [24], and MAX-TP-RCS-FR is reducible to MAX-CG. □

Sub-gradient based Algorithm: S. Ioannidis and E. Yeh [24] propose a distributed and adaptive algorithm for solving MAX-CG that converges to caching gain within $(1 - 1/e)$ factor from the optimal. The key idea of their algorithm is to first approximate MAX-CG with a convex optimization problem, as follows:

$$\text{Max:} \sum_{(i,s)\in\mathcal{R}} \lambda_{(i,s)} \sum_{k=1}^{|p|-1} w'_{p_{k+1}p_k}\min\{1, \sum_{k'=1}^{k} y_{p_{k'}i}\} \quad (27a)$$
$$\text{subj. to: } Y \in \mathcal{D}_1 \quad (27b)$$

where $\mathcal{D}_1$ represents the following constraints:

$$\sum_{i\in\mathcal{C}} y_{vi} \le C_v, \quad \text{for all } v \in V \setminus \mathcal{S}, \quad (28a)$$
$$y_{vi} \in [0, 1], \quad \text{for all } i \in \mathcal{C} \text{ and } v \in V, \quad (28b)$$
$$y_{vi} = 1, \quad \text{for all } i \in \mathcal{C} \text{ and } v \in \mathcal{S}_i \quad (28c)$$

They then approach the optimal solution by performing projected gradient ascend. As the objective function is not differentiable, a sub-gradient is used instead and estimated online. The desired content placement is obtained through state smoothening followed by a cache reshuffling step. Since MAX-TP-RCS-FR is reducible to MAX-CG, the same algorithm, under appropriate adaptations, can be used to solve MAX-TP-RCS-FR in a distributed and adaptive manner with the same optimality guarantees.

Recall that to transform MAX-TP-RCS-FR to MAX-CG, the weights of edges in MAX-CG needs to be properly configured according to Eqs. (25), while in the original MAX-CG problem they are fixed and are given a priori. Also observe that the network topology, routing paths and demands remain unchanged. This implies that the only challenge we need to address is how to correctly configure (virtual) weights of edges in a distributed and on-the-fly way, and make necessary adaptations in all computations involving weights (Luckily, as we can see later, the only adaptation lies in estimating sub-gradient as a function of packet loss rates and actual link weights). We elaborate the details below.

(1) As usual, each node v keeps as its state the probability that each content item $i \in \mathcal{C}$ is stored in cache, denoted by $y_{vi} \in [0, 1]$. Also denote by $k_p(v)$ the position of node v in path p, i.e., $p_l = v$ if $k_p(v) = l$.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

(2) Each request and data packet are associated with a control message (we call them request-control-message and data-control-message respectively, as depicted in Fig. 5), which is used to record packet loss rate over links that the message traverses. Data-control-message also includes weights of links as it flows.

(3) A request-control-message is generated whenever a request (i, s) is fed into the network and it flows in the same direction as the request. Once a request-control-message arrives at node v by traversing the link (u, v), the link loss rate f_{vu} and f_{uv} will be added by u or v in the message (we assume that these information is local to each node, i.e., each node maintains all its link states). The request-control-message is propagated over path $p = p_{(i,s)}$ until it passes a node u such that $\sum_{l=1}^{k_p(u)} y_{p_l i} > 1$, or u is a content server. Once this happens, a data-control-message is generated by u and all information (packet loss rates) recorded in the request-control-message are copied into the data-control-message.

(4) The data-control-message flows in reverse over the path. Once a data-control-message arrives at node v by traversing the link $(b, v) \in E$, the link weight w_{bv} is added in the message.

(5) Time is divided into intervals of length t. In every interval each node v on path p learns a quantity t_{vi} based on all the data-control-messages it receives:

$$t_{vi} = \frac{\sum_{k'=k_p(v)}^{|p|-1} w'^{(i,s)}_{p_{k'+1}p_{k'}} 1_{\sum_{l=1}^{k'} y_{p_l i} \le 1}}{\mathcal{N}_p(u, v)} \tag{29}$$

where

$$\mathcal{N}_p(u, v) = \prod_{l=1}^{k_p(u)-1} (1 - f_{p_l p_{l+1}}) \prod_{l=k_p(v)}^{k_p(o)-1} (1 - f_{p_{l+1} p_l}) \tag{30}$$

and u is the node generating the data-control-message. Note that with the information (packet loss rates and weights) contained in each data-control-message, t_{vi} can be computed online according to Eqs. (29) (30) (25).

(6) At the end of each interval of length t, each node v computes the following estimate for each content $i \in \mathcal{C}$:

$$z_{vi} = \frac{1}{t} \sum_{t \in \mathcal{T}_{vi}} t \tag{31}$$

where $\mathcal{T}_{vi}$ is the set of t_{vi}'s collected in a time interval. Node v then updates caching probability as follows:

$$y_{vi}^{k+1} \leftarrow \mathcal{P}_{\mathcal{D}_1}(y_{vi}^k + \eta_k z_{vi}) \tag{32}$$

where $\eta_k > 0$ is the step size and $\mathcal{P}_{\mathcal{D}_1}$ is the projection to domain $\mathcal{D}_1$.

Theorem 5: Let $z_v = [z_{vi}]_{i \in \mathcal{C}}$, $L(Y) = \sum_{(i,s)\in\mathcal{R}} \lambda(i, s) \sum_{k=1}^{|p|-1} w'^{(i,s)}_{p_{k+1}p_k} \min\{1, \sum_{k'=1}^{k} y_{p_{k'} i}\}$, then we have $\mathrm{E}[z_v(Y)] \in \partial_{y_v} L(Y)$, i.e., $\mathrm{E}[z_v(Y)]$ belongs to the sub-gradient set of $L(Y)$.

Proof: It has been shown in [24] that in problem MAX-CG we have $\mathrm{E}[z_v(Y)] \in \partial_{y_v} L(Y)$ when $t_{vi} = \sum_{k'=k_p(v)}^{|p|-1} w'^{(i,s)}_{p_{k'+1}p_{k'}} 1_{\sum_{l=1}^{k'} y_{p_l i} \le 1}$. The theorem follows by considering packet losses of control-messages. More specifically, note that when each request (i, s) from user is fed into the network, a request-control-message is generated. The probability that a corresponding data packet arrives at node v, denoted by $\mathcal{N}_p(u, v)$, is then given as Eq. (30). Dividing each measurement t_{vi} by this probability thus gives an unbiased estimator for a sub-gradient. □

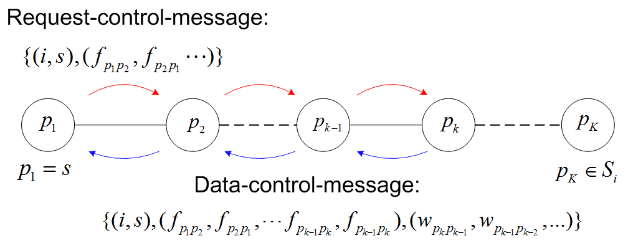

Request-control-message:
$\{(i,s),(f_{p_1p_2}, f_{p_2p_1} \cdots)\}$
p_1
p_2
p_{k-1}
p_k
p_K
$p_1 = s$
$p_K \in S_i$
Data-control-message:
$\{(i,s),(f_{p_1p_2}, f_{p_2p_1}, \cdots f_{p_{k-1}p_k}, f_{p_{k-1}p_k}),(w_{p_kp_{k-1}}, w_{p_{k-1}p_{k-2}}, \ldots)\}$

Fig. 5. A request-control-message and its corresponding data-control-message.

Similarly, we can prove this algorithm converges. All the other steps are keep unchanged and we therefore suggest readers refer to [24] for more details.

B. Potential-Based Adaptive Algorithm

The algorithm presented above has some drawbacks as caches are only updated at the end of each time interval. Moreover, cache reshuffling incurs additional routing cost for uncached items if they are to be cached. In this subsection, we propose a ***potential-based*** online algorithm that each node in network makes caching-decisions locally upon the arrival of individual content items.

Algorithm Overview: In the proposed algorithm, each node v maintains a quantity Q_{vi} for each item i, which we call *potential*. Q_{vi} is initialized as zero and is updated whenever a new request for item i or the corresponding data packet arrives, according to Rule 1 and Rule 2 given below. Node v then calculates caching probability for item i when it is to be cached and evicts items based on their potentials, according to Rule 3.

Rule 1: When a request (i, s) along path p arrives at node v, Q_{vi} is updated as follows:

$$Q_{vi} = Q_{vi} + \alpha \prod_{l=1}^{k_p(v)-1} (1 - f_{p_l p_{l+1}})(1 - f_{p_{l+1} p_l}) \tag{33}$$

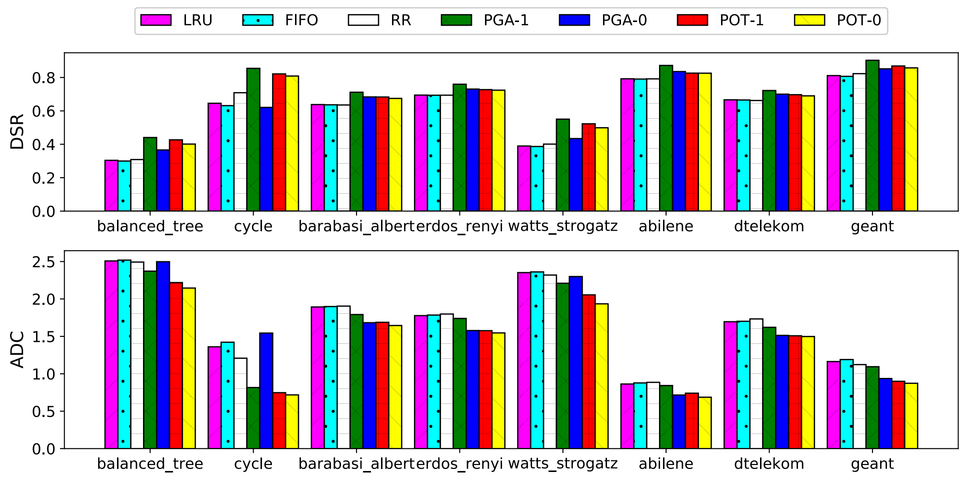
Rule 2: When a data packet i is generated by or arrives at node v along path p (in the reverse direction), Q_{vi} is updated accordingly:

$$Q_{vi} = Q_{vi} + (1 - \alpha)(\max_{u \in p} \mathrm{RT\_COST}(u) - \mathrm{RT\_COST}(v)) \tag{34}$$

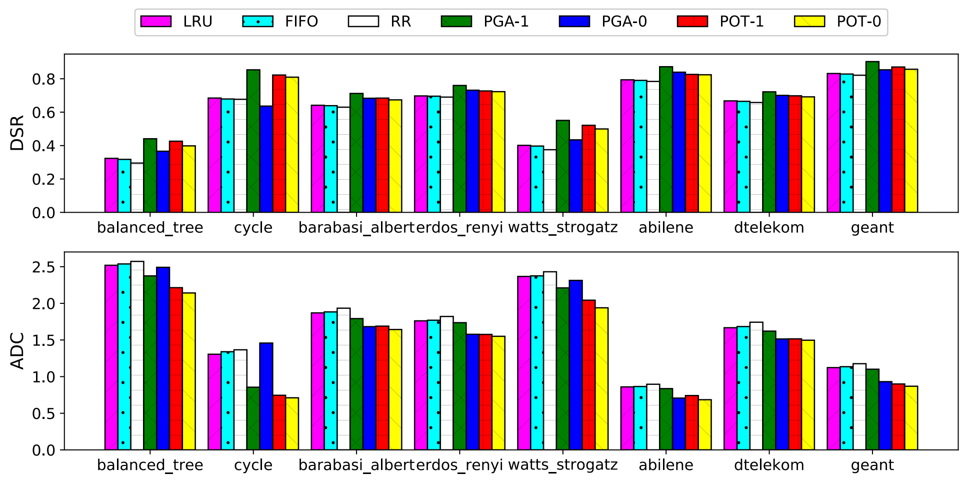
where $\mathrm{RT\_COST}(u)$ is defined as the *actual* routing cost if i is cached at u:

$$\mathrm{RT\_COST}(u) = \sum_{m=1}^{k_p(u)-1} w_{p_{m+1} p_m} \prod_{l=m+1}^{k_p(u)-1} (1 - f_{p_{l+1} p_l}) \tag{35}$$

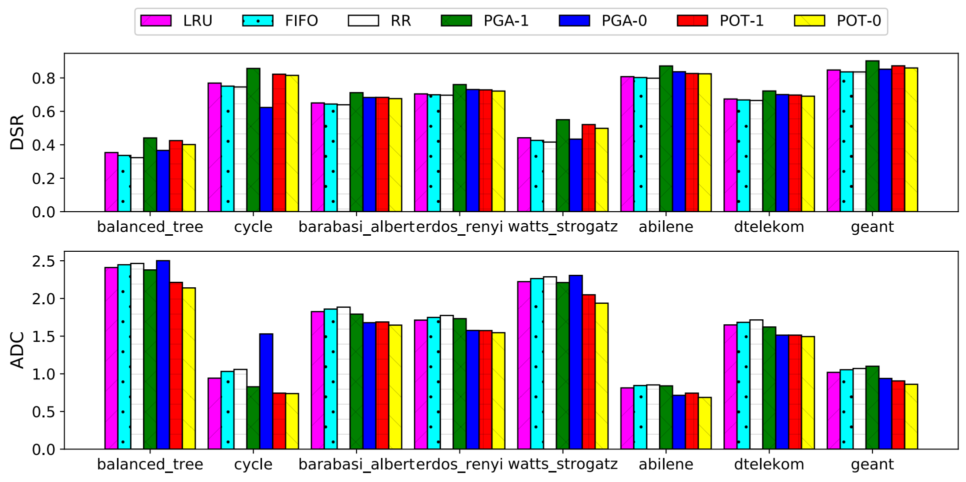
Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.



(a) LCE



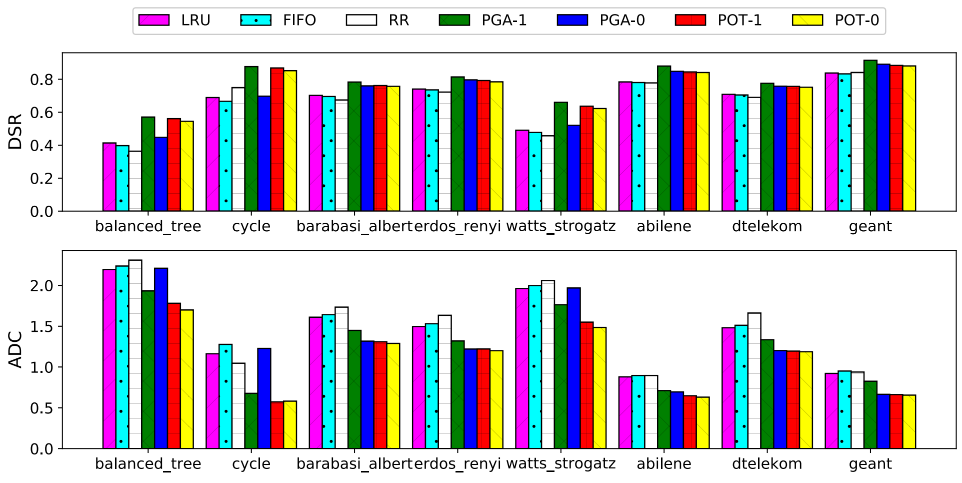
(b) LCD



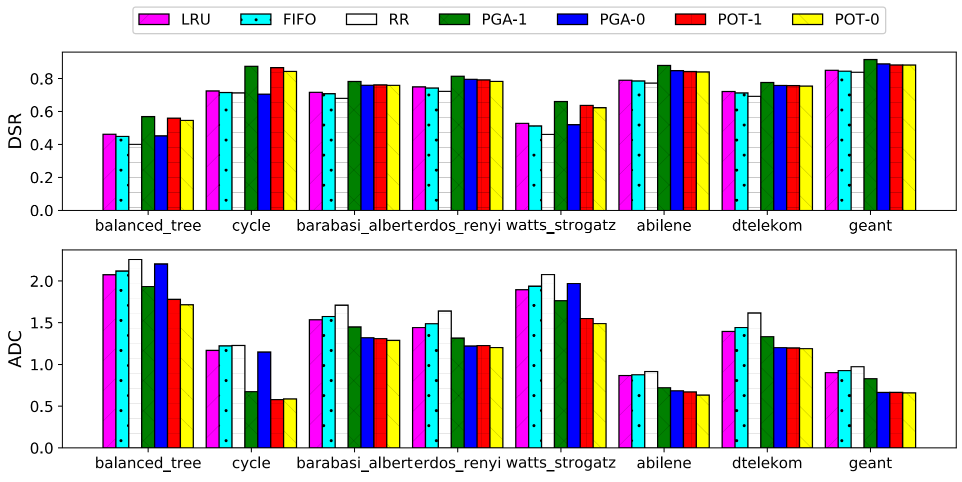
(c) ProbCache

Fig. 6. DSR and ADC for different topologies and caching algorithms under single-path routing scenario, with skewness parameter $\gamma = 0.8$.

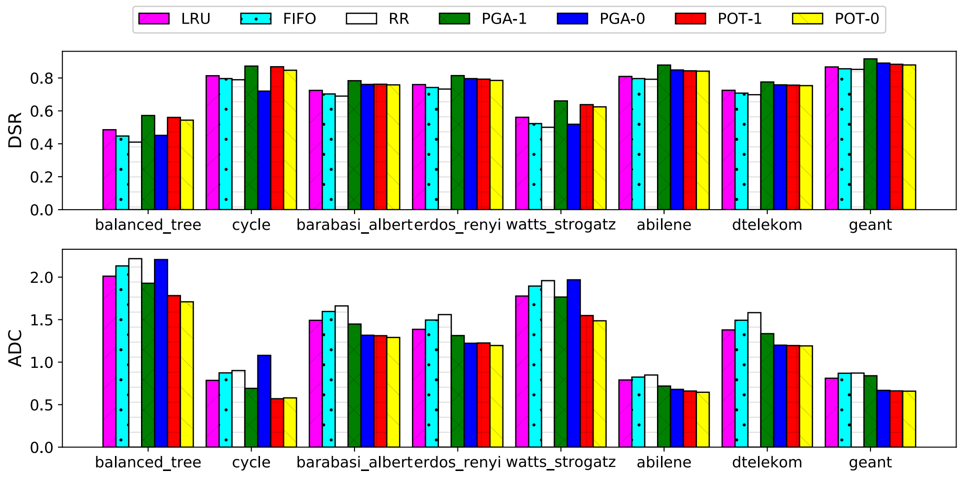
Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.



(a) LCE



(b) LCD



(c) ProbCache

Fig. 7. DSR and ADC for different topologies and caching algorithms under single-path routing scenario, with skewness parameter $\gamma = 1.2$.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

Rule 3: Whenever a new content item i arrives at node v and there's no room for it, v calculates the caching probability y_{vi} for i based on Q_{vi}'s:

$$y_{vi} = \frac{Q_{vi}}{Q_{vi} + \sum_{j \in \mathcal{C}_v} Q_{vj}} \tag{36}$$

where $\mathcal{C}_v$ is the set of items cached in v. If the decision is to cache i, then the item j with the least potential in node v, i.e., $j = \arg\min_{i \in \mathcal{C}_v} Q_{vi}$, is evicted.

Remark (key insights behind potential-based algorithm): 1) As each node updates potentials for each item whenever a request or a data packet flows, the more popular an item, the larger potential of it, and hence the larger caching probability. 2) Eq. (33) *captures the throughput gain if item i is cached at node v. As a result, it is preferable to cache items with large throughput. 3) Eq.* (34) *captures the actual routing cost gain*[5] *if i is cached at v. Therefore, the larger routing cost gain of an item, the larger caching probability for it.*

IV. Numerical Evaluation

In this section, we show the effectiveness of our proposed algorithm on improving system throughput and reducing network routing cost, and evaluate its performance by performing numerical simulations over a wide array of network topologies.

A. Evaluation Setup

We implement our potential-based online algorithm as well as the adapted projected gradient ascend algorithm on an open-source cache network simulator — `CacheNetwork` [1], and compare their performance with three traditional caching algorithms (LRU, FIFO and Random) and two recent proposals (MBP [44] and MAGIC [38]). We adopt three path replication policies — LCE (leave copy everywhere), LCD (leave copy down), and probabilistic caching (ProbCache, with caching probability 0.1). Both single-path and multi-path routing strategies are investigated. We consider the following two performance indexes.

(1) **Demand Satisfaction Ratio (DSR)**: This is the proportion of users' requests satisfied by network. Given demand and network topology, the larger DSR, the higher throughput, and vice versa.

(2) **Average Delivery Cost (ADC)**: This is the average content routing cost per request. Since RCS is hard to capture in network, we use this performance index to reflect the overall content routing cost in simulation.

Network Topologies: We use a wide array of network topologies for evaluation. Among them, `cycle`, `balanced_tree`, `barabasi_albert`, `erdos_renyi`, `watts_strogatz` are synthetic ones whereas `abilene`, `dtelekom`, `geant` represent real ones. The parameters of these graphs are chosen according to [25] (see section 7), and the weight of each edge is u.a.r (uniformly and randomly) selected from [1, 2]. Furthermore, each edge is configured as a high-loss-rate link with probability 30%, and low-loss-rate link with probability 70%. The packet loss rate of each low-loss-rate link and high-loss-rate link are u.a.r selected from [1%, 5%] and [25%, 30%], respectively [13]. Note that these settings represent networks with sever packet losses.

Workload Model: For each topology represented by graph $G(V, E)$, we associate a catalog of content items with size $|\mathcal{C}|$ and assume they are hosted by nodes randomly picked in V. We then randomly select a set $\mathcal{Q}$ of nodes from V as source nodes that generate a set $\mathcal{R}$ of requests with a Zipf distribution with skewness parameter γ. For single-path routing scenario, each request for a content item is routed over the shortest path to a content server. For multi-path routing scenario, we configure that there are 3 paths for each pair (*source node*, *content server*), and each request from *source node* to *content server* is randomly routed using one of these paths. The rate of each request is set as $\lambda_{(i,s)} = 1$ req/sec. Moreover, the sampling time for the projected gradient ascend algorithm is chosen as $T = 30$ sec, and the simulation time is 5000 sec. For fair comparison, all parameters of network topologies, cache sizes and simulation are set according to Table 2 in [25].

B. Simulation Results

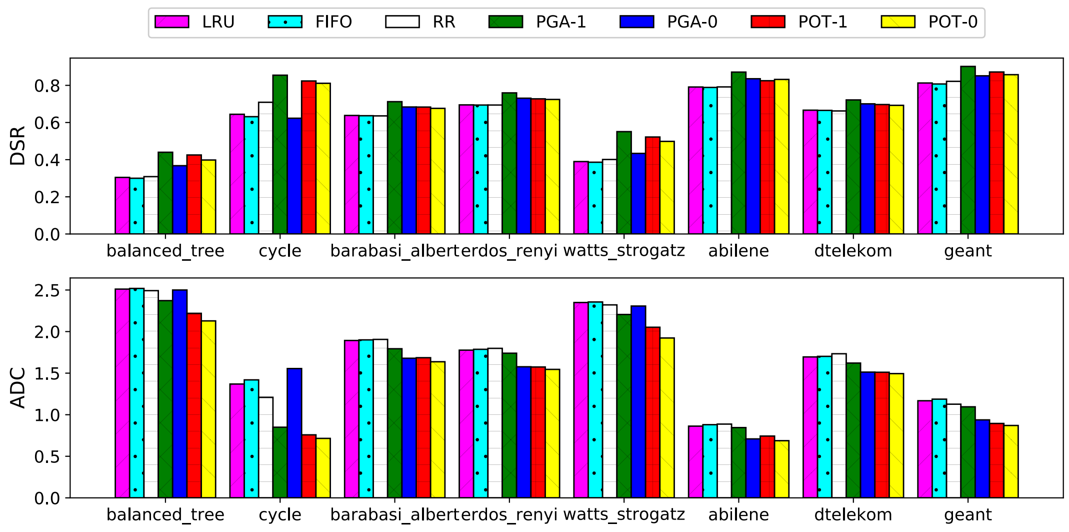
We first evaluate our proposed algorithm under single-path (shortest path) routing scenario against the three traditional caching algorithms. Figure 6 and Figure 7 show the two performance indexes for different topologies and caching algorithms, with different combination of path replication policies and Zipf skewness parameters ($\gamma = 0.8$ and $\gamma = 1.2$). Here PGA refers to the adapted projected gradient ascend algorithm. POT-1 stands for our potential-based online algorithm with $\alpha = 1$, i.e., when the goal is to solely maximize throughput, and POT-0 means $\alpha = 0$ with the goal being solely to optimize content routing cost. PGA-1 and PGA-0 can be interpreted in the same way.

From Figure 6 and Figure 7, we can see that as expected, our potential-based online algorithm outperforms the three traditional caching algorithms in DSR and ADC, with different path replication policies and skewness parameters, i.e., POT-1 achieves more throughput and POT-0 incurs lower content routing cost. However, while in most cases this happens to PGA, we observe that for some network topologies (`balanced_tree`, `cycle` and `watts_strogatz`), PGA-0 incurs higher content routing cost than the three traditional caching algorithms, i.e., it is observed that more than 50% of ADC is incurred by PGA-0 for topology `cycle` with $\gamma = 0.8$ and ProbCache, as compared to LRU. This indicates that in practice, our potential-based algorithm is more reliable and robust than PGA.

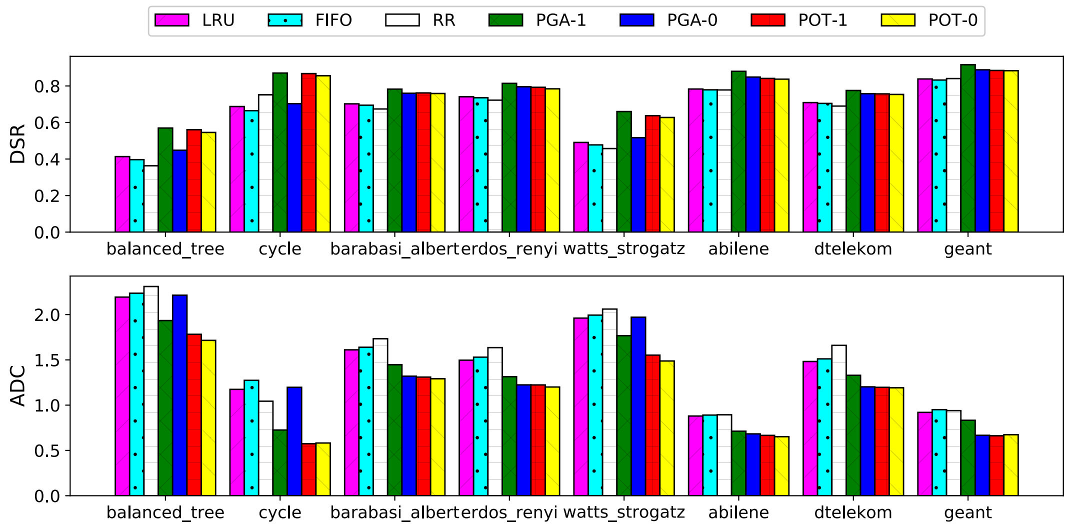
To fully understand how much improvement can be achieved by the two algorithms, we compare their performance indexes with that of the LRU, and the results are listed in Table II and Table III. Here for each algorithm and path replication policy, only the maximum and minimum improvement are listed (among the 8 topologies). From Table II ($\gamma = 0.8$), it can be seen that PGA-1 increases DSR by 4.8% (`dtelekom`, ProbCache) to 20.9% (`cycle`, LCE), and PGA-

[5]We define the routing cost gain as in Eq. (34) since routing cost under packet losses is no longer a monotone function of the node position along delivery path.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.



(a) $\gamma = 0.8$



(b) $\gamma = 1.2$

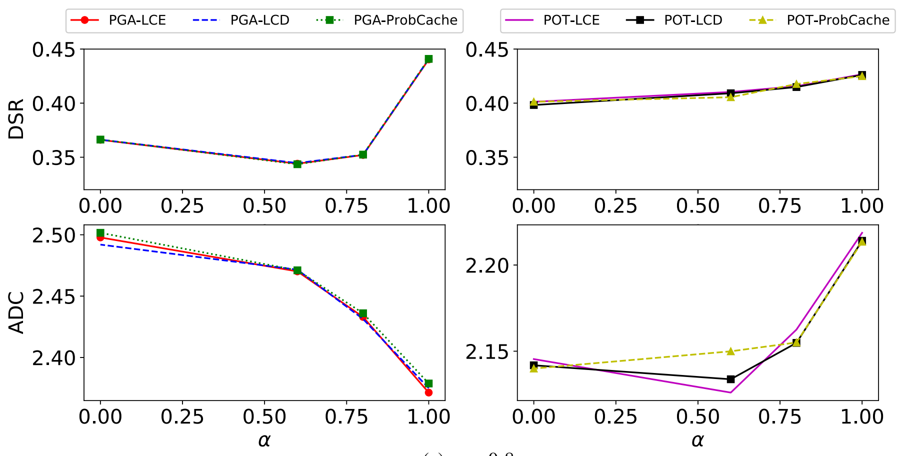
Fig. 8. DSR and ADC for different topologies and caching algorithms under multi-path routing scenario, with LCE path replication policy.

0 reduces ADC by -62.3% (`cycle`, ProbCache) to 17.1% (`geant`, LCD). On the other hand, POT-1 increases DSR by 1.9% (`abilene`, ProbCache) to 17.6% (`cycle`, LCE), and POT-0 reduces ADC by 11.2% (`balanced_tree`, ProbCache) to 47.2% (`cycle`, LCE). Similar results are observed in Table III with $\gamma = 1.2$. These results suggest that whereas the two algorithms have comparable capability in improving throughput, our potential-based online algorithm performs far more better in reducing content routing cost.

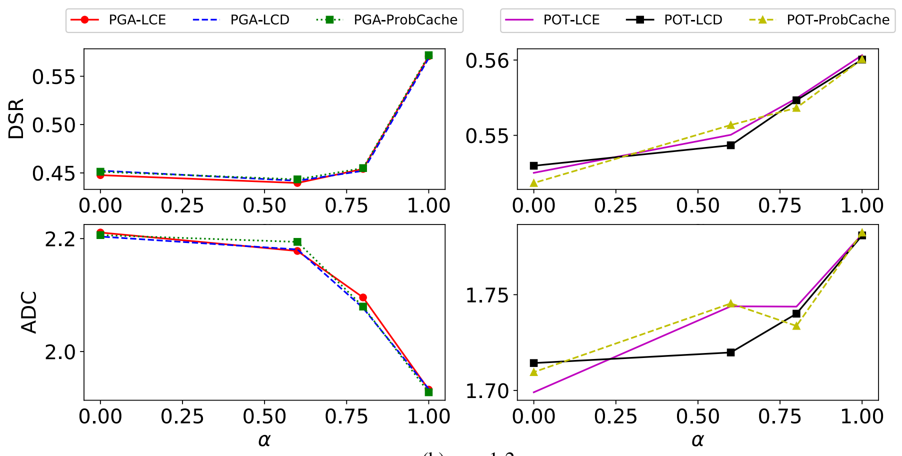
We also conduct simulations for multi-path routing scenario. For compactness and to avoid redundancy, only the performance with LCE and different skewness parameters are presented, as shown in Figure 8. Obviously, we can draw similar conclusions as that for the single-path routing scenario, i.e., both PGA-1 and POT-1 achieve more throughput than the three traditional caching algorithms, and POT-0 significantly reduces content routing cost for all network topologies. It follows that our potential-based algorithm is still effective under multi-path routing scenario.

Figure 9 shows how performance indexes of the two algorithms vary as the weight α increases. Interestingly, we find that for POT the performance tradeoff can be well controlled

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.



(a) $\gamma = 0.8$



(b) $\gamma = 1.2$

Fig. 9. Performance of the two algorithms under different α values.

TABLE II
PERFORMANCE IMPROVEMENT BY PGA AND OUR POTENTIAL-BASED ONLINE ALGORITHM WITH $\gamma = 0.8$

Path Replication Policy	DSR Improvement by PGA-1		DSR Improvement by POT-1		ADC Reduction by PGA-0		ADC Reduction by POT-0	
	Max	Min	Max	Min	Max	Min	Max	Min
LCE	20.9% (cycle)	5.5% (dtelekom)	17.6% (cycle)	3.2% (dtelekom)	19.5% (geant)	-13.5% (cycle)	47.2% (cycle)	14.4% (balanced_tree)
LCD	16.9% (cycle)	5.4% (dtelekom)	13.8% (cycle)	3.0% (erdos_renyi)	17.1% (geant)	-11.7% (cycle)	45.6% (cycle)	15.0% (balanced_tree)
ProbCache	10.8% (watts_strogatz)	4.8% (dtelekom)	7.9% (watts_strogatz)	1.9% (abilene)	8.2% (dtelekom)	-62.3% (cycle)	15.6% (abilene)	11.2% (balanced_tree)

by α. However, for PGA it is observed that increasing α improves both ADC and DSR, which is out of our expectation. We believe that this is in part due to the high correlation of the two performance indexes (consider for example, in a line network optimizing throughput will at the same time reduces content routing cost). Nevertheless, again we find that our potential-based algorithm is more reliable and robust.

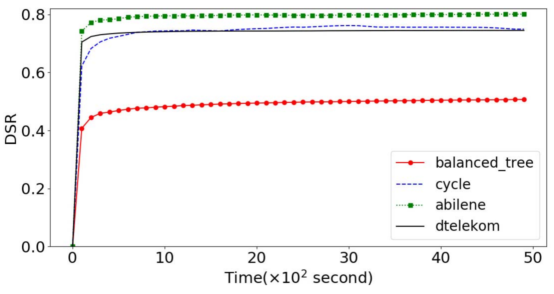
Figure 10 illustrates how the two algorithms behave as time goes on, where the path replication stragety is set as LCE and $\gamma = 1.2$. It can be seen that both algorithms converge fast that it takes approximately 100 sec to converge. Moreover, it can

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

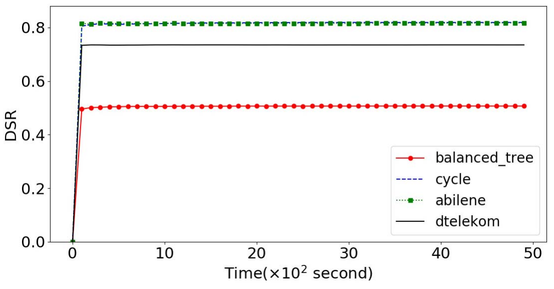
TABLE III

PERFORMANCE IMPROVEMENT BY PGA AND OUR POTENTIAL-BASED ONLINE ALGORITHM WITH $\gamma = 1.2$

Path Replication Policy	DSR Improvement by PGA-1		DSR Improvement by POT-1		ADC Reduction by PGA-0		ADC Reduction by POT-0	
	Max	Min	Max	Min	Max	Min	Max	Min
LCE	18.8% (cycle)	6.7% (dtelekom)	18.0% (cycle)	4.6% (geant)	27.8% (geant)	-5.7% (cycle)	49.9% (cycle)	22.6% (abilene)
LCD	14.9% (cycle)	5.5% (dtelekom)	14.1% (cycle)	3.2% (geant)	26.2% (geant)	-3.9% (cycle)	49.9% (cycle)	17.3% (balanced_tree)
ProbCache	9.9% (watts_strogatz)	4.9% (geant)	7.7% (watts_strogatz)	1.6% (geant)	17.6% (geant)	-37.5% (cycle)	18.7% (geant)	18.1% (balanced_tree)



(a) PGA



(b) POT

Fig. 10. Convergence time for the two algorithms with LCE and $\alpha = 0.8$.

be seen that our potential-based algorithm is more stable than PGA under different network topologies.

C. Comparison With Recent Proposals

Here we compare the performance of our potential-based online algorithm with MBP [44] and MAGIC [38], which were proposed in recent years. We adopt LCE path replication policy as all three algorithms are capable of dynamically determining the caching positions for each content item along the delivery path. From Fig. 11 and Fig. 12, we can see that among the three algorithms, POT-1 achieves the most throughput while POT-0 incurs the least routing cost, under both single-path and multi-path routing scenario, and with different network topologies and workload distributions. For example, under single-path routing scenario and with $\gamma = 1.2$, POT-1 increases DSR by 1.2% (`geant`) to 5.8% (`cycle`), while POT-0 reduces ADC by 6.4% (`dtelekom`) to 20.7% (`cycle`), as compared to MBP. When compared to MAGIC, the improvement for DSR is 0.3% (`dtelekom`) to 7.9% (`cycle`), and for ADC 4.8% (`erdos_renyi`) to 30% (`cycle`). Similar results can be observed under multi-path routing scenario.

D. Summary of Observations

We end this section by summarizing what we find in simulation. First, as compared to traditional caching algorithms as well as recent proposals, both PGA and POT are capable of improving network throughput and content routing cost under a variety of path replication policies and requests access patterns, for different network topologies. They have comparable performance of optimizing throughput, but POT performs far more better in optimizing content routing cost. Second, POT is more reliable and robust than PGA in that its performance tradeoff can be well adjusted by the weighting factor α. Moreover, it converges fast and behaves stable under different network topologies.

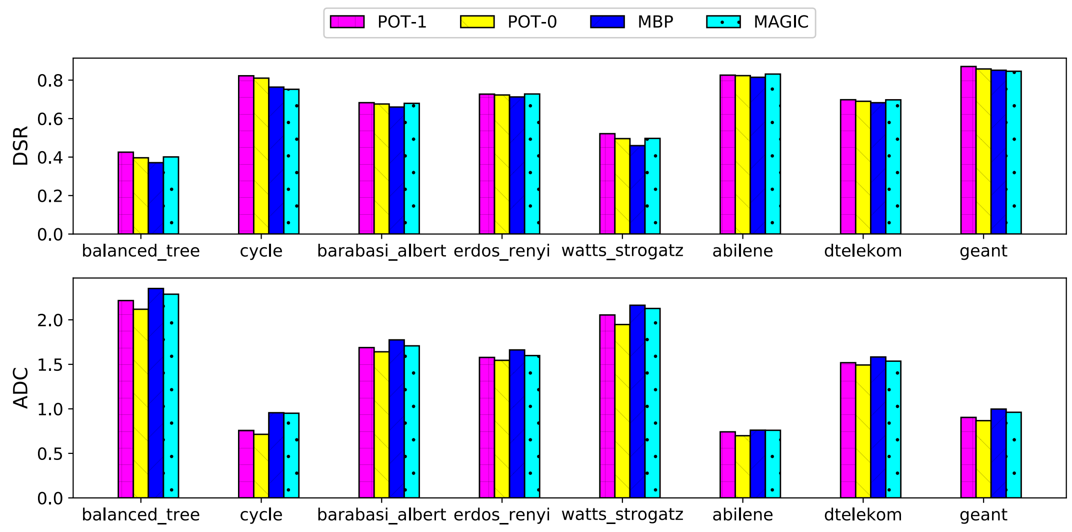
V. RELATED WORK

Performance evaluation and optimization for caching systems has long been a hot research topic. A significant amount of early work focused on modeling and analysis of various caching policies (i.e., LRU, FIFO, Random and more recently TTL-based [16], [21]). Both exact [8], [9] [20] and approximate models [21], [29] are provided to characterize cache performance under different workload models [6], [33] [23] and different network topologies [17], [32].

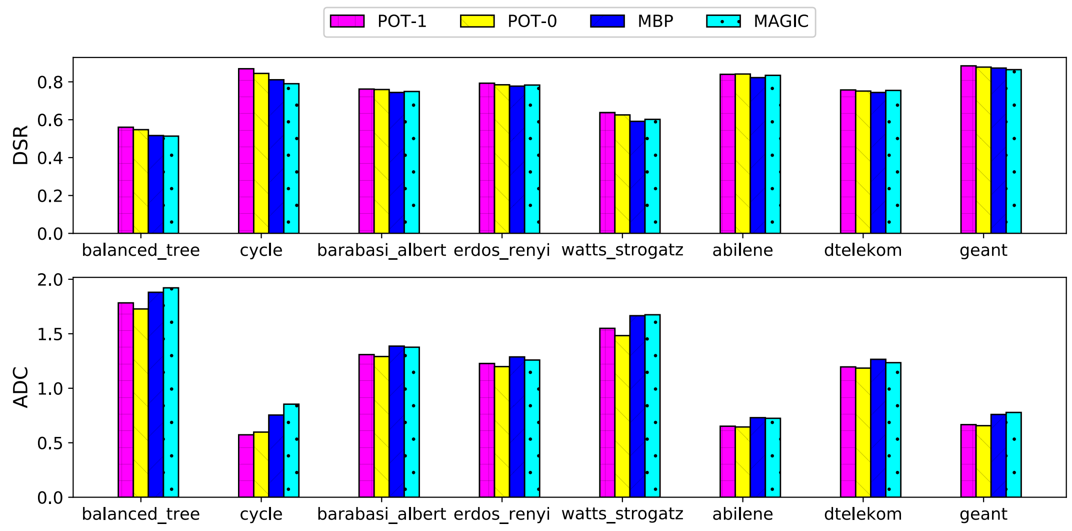
Both offline [11], [12] and online [14], [15] cache optimization problems have been investigated. Among them, online optimization of general cache networks, i.e., Information-Centric Networks [24], [31], has attracted particular research interests in recent years.

Our work belongs to online optimization of cache networks. The work that most close to our problem is [24], where authors consider optimizing network routing cost in a cache network by formulating a joint content placement and request routing optimization problem. Our work rely and expand upon

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.



(a) $\gamma = 0.8$



(b) $\gamma = 1.2$

Fig. 11. Performance comparison with two recent proposals (MBP and MAGIC) under single-path routing scenario, with LCE path replication policy.

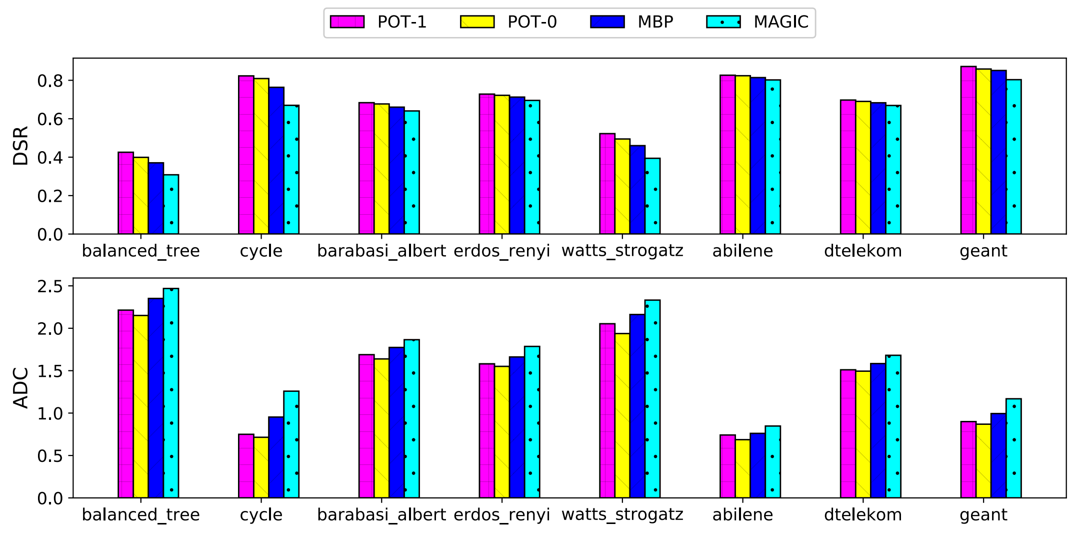
it by considering packet losses over links and taking network throughput as a new metric to optimize. The objective of our work is to maximize system throughput while maintaining the content routing cost for satisfied requests as low as possible. The consideration of throughput is novel, and the observation that throughput can be cast in the same framework as [24] is a very interesting result, which we believe it deserves to be reported to the community. The simple and adaptive algorithm based on “potentials” is also novel and interesting, and we believe this idea should contribute and inspire development of future caching algorithms.

VI. Conclusion and Future Work

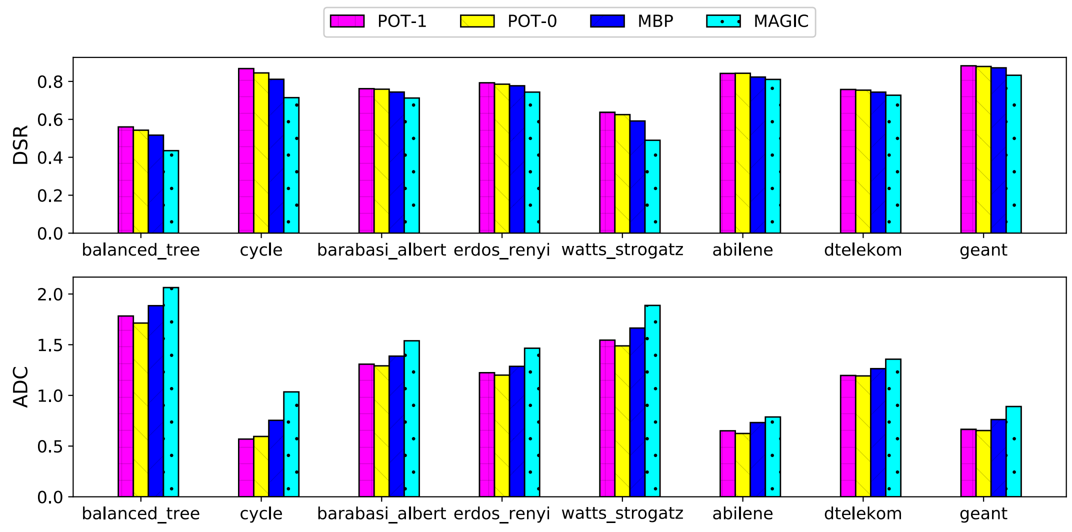
A. Conclusion

In this paper, we study the problem of optimizing content delivery over a lossy cache network, taking both throughput and content routing cost into account. We propose a new performance metric – RCS, to reflect the actual routing cost in content delivery while at the same time attains attractive mathematical properties such as super-modularity. We then formulate an optimization problem for the task under fixed routing scenario, and prove it is reducible to the one of minimizing

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.



(a) $\gamma = 0.8$



(b) $\gamma = 1.2$

Fig. 12. Performance comparison with two recent proposals (MBP and MAGIC) under multi-path routing scenario, with LCE path replication policy.

content routing cost without packet losses. We present necessary rules for the reduction, and leverage existing distributed and adaptive algorithm to solve the problem. A simple and adaptive online algorithm is further proposed. Last, we evaluate our models and algorithms through simulations over a wide array of network topologies.

B. Future Work

There are several directions for future research. First, while in this work we propose RCS to capture both the heterogeneous routing cost and packet loss rates over links, an interesting problem is whether there exists alternative metrics that can better reflect the actual routing cost for delivered content, which also possess nice mathematical properties that leads to efficient algorithm design. Second, the joint optimization problem under dynamic routing scenario, i.e., when the path to deliver each request is dynamically selected based on the optimization objective and system workload, remains unexplored. It is also an open problem as how to develop efficient online/distributed caching and routing algorithm for this problem. Finally, in lemma 2.2 we have shown that optimizing RCS and throughput are two different goals and sometimes they can even conflict. However, we also observed in simulation that they are

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

sometimes highly correlated, i.e., for the algorithm PGA maximizing throughput at the same time leads to a low RCS. It remains unclear as why there is such a high correlation, and under what conditions does the two metrics exhibit such correlation, and then how to characterize it? These problems are no doubt important and worthy of further investigation.

REFERENCES

[1] *Cachenetwork*. Accessed: Jun. 16, 2019. [Online]. Available: https://github.com/neu-spiral/CacheNetwork

[2] *Named Data Networking*. Accessed: Sep. 16, 2020. [Online]. Available: https://named-data.net/

[3] "Cisco visual networking index: Forecast and trends, 2017–2022," Cisco, San Francisco, CA, USA, White Paper, 2019. [Online]. Available: https://www.entersoftware.it/wp-content/uploads/2020/02/Cisco-Visual-Network-Index_Forecast-2017-22.pdf

[4] A. A. Ageev and M. I. Sviridenko, "Pipage rounding: A new method of constructing algorithms with proven performance guarantee," *J. Combinat. Optim.*, vol. 8, no. 3, pp. 307–328, Sep. 2004.

[5] B. Ahlgren, C. Dannewitz, C. Imbrenda, D. Kutscher, and B. Ohlman, "A survey of information-centric networking," *IEEE Commun. Mag.*, vol. 50, no. 7, pp. 26–36, Jul. 2012.

[6] S. M. Azimi, O. Simeone, A. Sengupta, and R. Tandon, "Online edge caching and wireless delivery in fog-aided networks with dynamic content popularity," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1189–1202, Jun. 2018.

[7] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, and M. Wählisch, "Information centric networking in the iot: Experiments with ndn in the wild," in *Proc. 1st ACM Conf. Inf.-Centric Netw.*, 2014, pp. 77–86.

[8] D. S. Berger, P. Gland, S. Singla, and F. Ciucu, "Exact analysis of TTL cache networks," *Perform. Eval.*, vol. 79, pp. 2–23, Sep. 2014.

[9] D. S. Berger, P. Gland, S. Singla, and F. Ciucu, "Exact analysis of TTL cache networks: The case of caching policies driven by stopping times," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 42, no. 1, pp. 595–596, 2014.

[10] G. Carofiglio, V. Gehlen, and D. Perino, "Experimental evaluation of memory management in content-centric networking," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2011, pp. 1–6.

[11] W. Chu, M. Dehghan, J. C. S. Lui, D. Towsley, and Z.-L. Zhang, "Joint cache resource allocation and request routing for in-network caching services," *Comput. Netw.*, vol. 131, pp. 1–14, Feb. 2018.

[12] W. Chu, M. Dehghan, D. Towsley, and Z.-L. Zhang, "On allocating cache resources to content providers," in *Proc. 3rd ACM Conf. Inf.-Centric Netw.*, Sep. 2016, pp. 154–159.

[13] W. Chu, X. Guan, Z. Cai, and L. Gao, "Real-time volume control, for interactive network traffic replay," *Comput. Netw.*, vol. 57, no. 7, pp. 1611–1629, 2013.

[14] M. Dehghan, W. Chu, P. Nain, D. Towsley, and Z.-L. Zhang, "Sharing cache resources among content providers: A utility-based approach," *IEEE/ACM Trans. Netw.*, vol. 27, no. 2, pp. 477–490, Apr. 2019.

[15] M. Dehghan, L. Massoulie, D. Towsley, D. S. Menasche, and Y. C. Tay, "A utility optimization approach to network cache design," *IEEE/ACM Trans. Netw.*, vol. 27, no. 3, pp. 1013–1027, Jun. 2019.

[16] C. F. N. Eric, N. Philippe, N. Giovanni, and T. Don, "Analysis of TTL-based cache networks," in *Proc. 6th Int. Conf. Perform. Eval. Methodologies Tools*, 2012, pp. 1–10.

[17] N. C. Fofack, P. Nain, G. Neglia, and D. Towsley, "Performance evaluation of hierarchical TTL-based cache networks," *Comput. Netw.*, vol. 65, pp. 212–231, Jun. 2014.

[18] C. Fricker, P. Robert, and J. Roberts, "A versatile and accurate approximation for LRU cache performance," in *Proc. 24th Int. Teletraffic Congr.*, 2012, p. 8.

[19] M. Garetto, E. Leonardi, and V. Martina, "A unified approach to the performance analysis of caching systems," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 1, no. 3, p. 12, 2016.

[20] N. Gast and B. Van Houdt, "Asymptotically exact TTL-approximations of the cache replacement algorithms LRU (M) and H-LRU," in *Proc. 28th Int. Teletraffic Congr. (ITC)*, vol. 1, 2016, pp. 157–165.

[21] N. Gast and B. Van Houdt, "TTL approximations of the cache replacement algorithms LRU(m) and h-LRU," *Perform. Eval.*, vol. 117, pp. 33–57, Dec. 2017.

[22] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, Sep. 2013.

[23] P. Hassanzadeh, A. M. Tulino, J. Llorca, and E. Erkip, "On coding for cache-aided delivery of dynamic correlated content," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 8, pp. 1666–1681, Aug. 2018.

[24] S. Ioannidis and E. Yeh, "Adaptive caching networks with optimality guarantees," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 44, pp. 113–124, Jun. 2016.

[25] S. Ioannidis and E. Yeh, "Jointly optimal routing and caching for arbitrary network topologies," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1258–1275, Jun. 2018.

[26] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. 5th Int. Conf. Emerg. Netw. Exp. Technol.*, 2009, pp. 1–12.

[27] P. Janis *et al.*, "Device-to-device communication underlaying cellular communications systems," *Int. J. Commun., Netw. Syst. Sci.*, vol. 2, no. 3, p. 169, 2009.

[28] M. Ji, G. Caire, and A. F. Molisch, "The throughput-outage tradeoff of wireless one-hop caching networks," *IEEE Trans. Inf. Theory*, vol. 61, no. 12, pp. 6833–6859, Dec. 2015.

[29] B. Jiang, P. Nain, and D. Towsley, "On the convergence of the TTL approximation for an LRU cache under independent stationary request processes," *ACM Trans. Model. Perform. Eval. Comput. Syst.*, vol. 3, no. 4, pp. 1–31, Sep. 2018.

[30] I. Lee and K. Lee, "The Internet of Things (IoT): Applications, investments, and challenges for enterprises," *Bus. Horizons*, vol. 58, no. 4, pp. 431–440, Jul. 2015.

[31] J. Li *et al.*, "DR-cache: Distributed resilient caching with latency guarantees," in *Proc. INFOCOM IEEE Conf. Comput. Commun.*, Apr. 2018, pp. 441–449.

[32] K. Li, C. Yang, Z. Chen, and M. Tao, "Optimization and analysis of probabilistic caching in N-tier heterogeneous networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 2, pp. 1283–1297, Feb. 2018.

[33] Y. Lu, W. Chen, and H. V. Poor, "Coded joint pushing and caching with asynchronous user requests," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 8, pp. 1843–1856, Aug. 2018.

[34] M. Mahdian, A. Moharrer, S. Ioannidis, and E. Yeh, "Kelly cache networks," in *Proc. INFOCOM IEEE Conf. Comput. Commun.*, Apr. 2019, pp. 217–225.

[35] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C.-N. Chuah, Y. Ganjali, and C. Diot, "Characterization of failures in an operational IP backbone network," *IEEE/ACM Trans. Netw.*, vol. 16, no. 4, pp. 749–762, Aug. 2008.

[36] L. Muscariello, G. Carofiglio, and M. Gallo, "Bandwidth and storage sharing performance in information centric networking," in *Proc. ACM SIGCOMM Workshop Inf.-Centric Netw. (ICN)*, 2011, pp. 26–31.

[37] G. S. Paschos, G. Iosifidis, M. Tao, D. Towsley, and G. Caire, "The role of caching in future communication systems and networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1111–1125, Jun. 2018.

[38] J. Ren *et al.*, "MAGIC: A distributed MAx-gain in-network caching strategy in information-centric networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2014, pp. 470–475.

[39] M. Satyanarayanan, "The emergence of edge computing," *Computer*, vol. 50, no. 1, pp. 30–39, Jan. 2017.

[40] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Comput.*, vol. 8, no. 4, pp. 14–23, Oct. 2009.

[41] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "FemtoCaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 8402–8413, Dec. 2013.

[42] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[43] C.-X. Wang *et al.*, "Cellular architecture and key technologies for 5G wireless communication networks," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 122–130, Feb. 2014.

[44] H. Wu, J. Li, and J. Zhi, "MBP: A max-benefit probability-based caching strategy in information-centric networking," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 5646–5651.

[45] J. Yao and N. Ansari, "Joint content placement and storage allocation in C-RANs for IoT sensing service," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 1060–1067, Feb. 2019.

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

[47] L. Zhang *et al.*, "Named data networking," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 3, pp. 66–73, 2014.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.

Weibo Chu received the B.S. degree in software engineering and the Ph.D. degree in control science and engineering from Xi'an Jiaotong University, Xi'an, China, in 2005 and 2013, respectively. From 2011 to 2012, he worked as a Visiting Researcher with Microsoft Research Asia, Beijing. Since 2013, he has been with the School of Computer Science and Technology, Northwestern Polytechnical University, where he is currently an Associate Professor. He has participated in various research and development projects on network testing, performance evaluation, and troubleshooting, and gained extensive experiences in the development of networked systems for research and engineering purposes. His research interests include internet measurement and modeling, traffic analysis, and performance evaluation.

John C. S. Lui (Fellow, IEEE) received the Ph.D. degree in computer science from UCLA. He was the Chairman of the Department of Computer Science and Engineering from 2005 to 2011. He is currently a Professor with the Department of Computer Science and Engineering, The Chinese University of Hong Kong. His current research interests include communication networks, network/system security (e.g., cloud security and mobile security), network economics, network sciences (e.g., online social networks and information spreading), cloud computing, large-scale distributed systems, and performance evaluation theory. He is an Elected Member of the IFIP WG 7.3, a fellow of the ACM, and a Croucher Senior Research Fellowship. He received various departmental teaching awards and the CUHK Vice-Chancellor's Exemplary Teaching Award. He was also a co-recipient of the IFIP WG 7.3 Performance 2005 and the IEEE/IFIP NOMS 2006 Best Student Paper Award. He serves on the Editorial Board for IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, *Journal of Performance Evaluation*, and *International Journal of Network Security*.

Zhiwen Yu (Senior Member, IEEE) received the Ph.D. degree in computer science from Northwestern Polytechnical University, Xi'an, China, in 2005. He was an Alexander Von Humboldt Fellow with Mannheim University, Germany, and a Research Fellow with Kyoto University, Kyoto, Japan. He is currently a Professor and the Dean of the School of Computer Science, Northwestern Polytechnical University. His research interests include ubiquitous computing and mobile computing.

Yi Lin received the Ph.D. degree in computer science from Northwestern Polytechnical University, Xi'an, China, in 2005. He is currently an Associate Professor with the School of Computer Science, Northwestern Polytechnical University. His research interests include data storage and software engineering.

Authorized licensed use limited to: Chinese University of Hong Kong. Downloaded on July 24,2021 at 07:04:14 UTC from IEEE Xplore. Restrictions apply.