Computer Networks 226 (2023) 109654

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Privacy-preserving edge caching: A probabilistic approach

Seyedeh Bahereh Hassanpour [a], Ahmad Khonsari [a,b,*], Masoumeh Moradian [b], Seyed Pooya Shariatpanahi [a]

[a] School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran
[b] School of Computer, Institute for Research in Fundamental Science (IPM), Tehran, Iran

ARTICLE INFO

Keywords:
Edge cache network
Privacy
Chunk-based probabilistic caching
Communication cost
Linear optimization

ABSTRACT

Edge caching (EC) decreases the average access delay of end-users through caching popular content at the edge of the network, however, it increases the leakage probability of valuable information such as users' preferences. Most of the existing privacy-preserving approaches focus on adding extra layers of encryption, which confronts the network with more challenges such as energy and computation limitations. We employ a chunk-based joint probabilistic caching (JPC) approach to mislead an adversary eavesdropping on the communication inside an EC network and maximizing the adversary's error in estimating the requested file and the requesting cache. Then, we optimize the probability of different cache placements in JPC in order to minimize the communication cost while guaranteeing the desired privacy. We show that the optimization problem can be formulated as a linear programming (LP) problem. Since JPC inherits the curse of dimensionality, we also propose scalable JPC (SPC), which reduces the number of feasible cache placements by dividing the files into non-overlapping subsets. We also compare the JPC and SPC approaches against an existing probabilistic method, referred to as disjoint probabilistic caching (DPC), as well as a random dummy-based approach (RDA). Results obtained through extensive numerical evaluations confirm the validity of the analytical approach and the superiority of JPC and SPC over DPC and RDA.

1. Introduction

The deluge of interest in using delay-critical Internet-of-things (IoT) applications and delay-sensitive data services, even beyond fifth-generation (B5G) communications, poses a significant challenge to the data processing capabilities of the network. In this regard, the edge network emerged as a pivotal key in 5G for providing computation, storage, and processing much closer to the end-users compared to the cloud networks. Edge computing, as a paradigm in edge networks, provides service delivery at the edge of the network and mitigates transmitting the computation to more distant servers inside the cloud through equipping the intermediate servers with edge nodes such as micro base stations or WiFi access points [1,2]. Furthermore, edge caching (EC), as a promising technique to store popular content closer to the end-users, decreases the traffic of the backhaul links and improves the quality of experience (QoE) by the end-users in terms of access delay [3,4].

EC paradigm enhances security and privacy by bringing the content closer to the end-users and eliminating the access of multiple intermediate nodes to the data. However, this geographical proximity also brings the potential (active/passive) attackers closer to the critical/personal information such as users' location or personal preferences. Therefore, the network is more vulnerable to different types of attacks [5,6]. For example, an attacker may interrupt the communication between the user and the caching edge device through a jamming attack, break down an EC server through a distributed denial of service (DoS) attack, or get access to the caching contents and network resources as a fake legal user through spoofing attacks [7,8]. On the other hand, the EC network consists of distributed edge devices controlled by autonomous people or companies. These owners may be curious about the data contents stored on their caches and even launch insider attacks or eavesdrop to obtain critical private information of the customers and sell them for different purposes, e.g., to the advertisements companies [7,9]. Generally, the main motivation of privacy attacks in EC networks is to derive the identity of the requesting users, their queries, and the statistics of the queries, e.g., the popularity of the contents.

Much of the focus of researchers in recent years has been on studying the location [10–12] and the pattern privacies [13], which aim to secure the location and the usage pattern of the users, respectively. The proposed solutions to tackle the privacy issues exploit cryptography and anonymity [14–17], information-theory [18,19], machine-learning [7,

* Corresponding author.
E-mail addresses: b.hassanpour@ut.ac.ir (S.B. Hassanpour), a_khonsari@ut.ac.ir (A. Khonsari), mmoradian@ipm.ir (M. Moradian), p.shariatpanahi@ut.ac.ir (S.P. Shariatpanahi).

https://doi.org/10.1016/j.comnet.2023.109654
Received 18 March 2022; Received in revised form 1 January 2023; Accepted 18 February 2023
Available online 28 February 2023

20], and dummy transmissions [12]. In [14], the authors propose a pseudonyms-based approach to conceal the real identity of the contents belonging to the content providers (CPs) and users' requests from the ISP, which is the cache owner, in a content distribution network (CDN). Due to the fact that over time ISP can discover the relation between the fake and real ID of the contents, there is a need to refresh the encrypted name. For this purpose, the authors in [15] derive the optimal number of encryption refreshes in a static time (e.g., a day). In [16], the authors preserve the privacy of CPs and users from ISP by using Shamir secret sharing (SSS), which shares the content popularity among caches without letting the ISP know.

Despite the abundance of cryptographic solutions, these solutions mostly suffer from computation complexity, power consumption, and decryption delays. However, the security and privacy-preserving solutions at the edge should possess low complexity due to the limited computational power and memory of the edge nodes as well as the energy and hardware limitations. In this regard, the authors in [18] propose a coding scheme that includes SSS and replicated subtasks to provide information-theoretic data privacy in the presence of untrustworthy edge servers. The information-theoretic approach is also applied in [19], to maximize a lower bound of adversary's estimation error derived from Fano inequality. Then, an ϵ-constraint optimization is proposed to find the caching probability of the files maximizing the proposed lower bound. In the above studies, the popularities of contents are known. In the case of unknown or time-varying popularities, machine learning techniques are mostly employed to learn popularities, which suffer from privacy leakage since they require sharing the requests' information with a central node for the aim of training. Thus, federated learning is proposed to overcome the privacy issues in online learning scenarios [20,21]. Another general policy in preserving privacy in the presence of eavesdroppers is the dummy-transmission-based approach, which relies on transmitting dummy queries by the caches or dummy information by the server [12] in the content delivery phase in order to obfuscate the eavesdropper. The dummy-based approaches suffer from backhaul traffic increase due to extra unnecessary transmissions. Probabilistic caching can also be employed in an EC network to leverage the privacy degree of the network since it increases the ambiguity of the adversaries over the caches' contents and thus, increases the privacy degree of the network. Furthermore, it takes advantage of lower complexity compared to cryptographic methods. Probabilistic cache placement is employed in [22] to provide physical-layer security in a wireless cache-aided network in the presence of eavesdroppers. They optimize the probability of caching individual files to maximize the number of transmissions not decoded by eavesdroppers while minimizing the communication cost.

In this paper, we study probabilistic caching to preserve the desired privacy in an EC network while minimizing the network communication cost. In particular, our proposed EC network consists of a single server and K distributed edge caches, where the server is in charge of content placement in the caches and delivering uncached contents to them. Furthermore, a passive eavesdropper monitors the amount of traffic transferred over the shared link between the server and the caches. In the proposed scenario, we minimize the communication cost while satisfying a minimum privacy degree, where the communication cost and privacy degree are defined as the average amount of traffic over the shared link and the error probability of the adversary, respectively. We assume that the adversary is aware of the content popularities and probabilistic caching protocols [23]. However, it has no access to the information of cache queries and their corresponding responses. As such, the adversary can only measure the amount of transferred traffic over the shared link in the content delivery (CD) phase and thus, exploits this information to estimate the identity of the requests. Unlike the previous studies [24,25], which optimize the probability of caching individual files in order to satisfy the desired performance metrics, we optimize the probabilities of joint placements of the files in the caches and highlight its advantages throughout the paper. The main contributions of the paper are as follows.

- We define joint probabilistic caching (JPC) policy rigorously and formulate the optimization of communication cost constrained to a minimum guaranteed privacy under the JPC approach. Then, we show that the proposed optimization can be written as a Linear Programming (LP) problem. We also propose the hit-ratio-based optimization of JPC and assert that chunk-based optimization provides more flexibility for achieving higher cache hit ratios.
- We solve the same optimization problem as in JPC considering the disjoint probabilistic caching (DPC) policy, in which the probabilities of caching individual files are optimized instead of the probabilities of different cache placements. Also, we show that optimal DPC has the same performance as optimized non-chunked-based JPC. However, chunk-based JPC outperforms DPC when hit-ratio constraints are required.
- We propose scalable JPC, in which the complexity decreases compared to JPC through caching chunks chosen from L non-overlapping subsets of files instead of N files. The performance of scalable JPC can be arbitrarily close to the optimal performance of JPC through increasing the number of subsets L.
- Finally, we present extensive numerical and simulation results to validate our analytical approach. We also propose a random dummy approach as a benchmark and show that JPC and scalable JPC outperform the dummy approach significantly. Also, we show that by choosing proper subsets in scalable JPC, we can shrink the feasible set in the corresponding LP optimization significantly while keeping the performance very close to the optimal performance of JPC.

The remainder of the paper is organized as follows. In Section 2, we describe the EC network, the adversary model, and the performance metrics. Section 3, provides the problem formulation and LP optimization of JPC. Section 4 is dedicated to DPC optimization and its comparison against JPC. Section 5 presents scalable JPC method. Numerical results are presented in Section 8. Finally, Section 9 concludes the paper.

2. System model and assumptions

In this section, we introduce our system model, an EC network comprised of an edge server node and distributed caches and an adversary eavesdropping on transmissions within the caching network. Then, we introduce the related performance metrics, including communication cost and privacy degree, and give an example to clarify the trade-off between these two metrics. Table 1 lists the notations used in this paper.

2.1. Edge caching model

As depicted in Fig. 1, our proposed caching network consists of a single edge server node, referred to as the server hereafter, and a set of distributed edge caches, denoted by $\mathcal{K} = \{1, 2, \dots, K\}$, that are connected to the server through a shared link. The server has full access to a library consisting of N files of equal size, i.e., $\mathcal{N} = \{1, 2, \dots, N\}$, each of which is partitioned into C chunks. Moreover, each cache $k \in \mathcal{K}$ is capable of caching M files, or equivalently MC chunks. Furthermore, each cache serves a distinct set of end-users. It is worth noting that if the number of end-users at each cache is equal to one, the model corresponds to the case where the distributed caches are located at end-users. Otherwise, the edge caches model the distributed caches owned by an ISP. Nevertheless, both cases can be applied in our scenario since as will be discussed, we focus on the privacy of transmissions between the caches and the server, which is independent of the number of users under the coverage of caches.

In compliance with the convention, the caching process in our considered scenario is performed in two phases; cache content placement

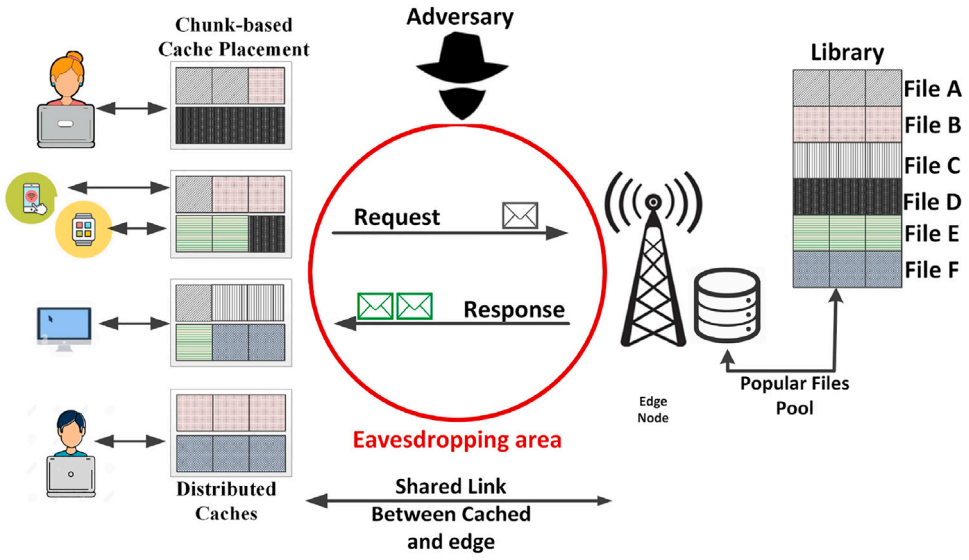


Fig. 1. The adversarial system model with an edge server which is responsible to serve the caches through a shared link.

(CCP) and content delivery (CD). In the CCP phase, the server fills up the cache's storage with chunks chosen from different files. In our analysis, all chunks of a file are equally important, and thus, we only focus on the number of chunks cached from a file, regardless of which exact chunks have been cached. Moreover, the total number of chunks in each cache does not exceed the capacity of the cache, i.e., MC. Consequently, let $\mathbf{z} = (z_1, z_2, \dots, z_N)$ denote a feasible chunk placement at a typical edge cache, where z_i denotes the number of chunks stored from file i. Then, regarding that all caches are identical, the set of feasible chunk placements at each edge cache, represented by $\mathcal{F}$, is written as

$$\mathcal{F} = \{\mathbf{z} | 0 \le z_i \le C, i \in \{1, \dots, N\}, \sum_{i=1}^{N} z_i = MC\}. \quad (1)$$

We assume that a central entity decides about the contents of the caches in the CCP phase. Without loss of generality, we consider the server as a decision-maker in our scenario, e.g., the server plays the role of a content provider in the real world. In particular, the server chooses the placement of each cache according to a probabilistic caching policy, where the policy in fact, indicates a probability distribution over all feasible cache placements, i.e., $\mathcal{F}$. Thus, it is evident that in the proposed probabilistic caching, the chunk places in each cache are filled jointly rather than independently. In this regard, we define the joint probabilistic caching policy more rigorously as follows.

Definition 1. (Joint Probabilistic Caching Policy) The joint probabilistic caching (JPC) policy at cache k is defined as a probability distribution over $\mathcal{F}$, denoted by $P^{(k)}(\mathbf{z})$. Let $\mathbf{Z}^{(k)} = (Z_1^{(k)}, Z_2^{(k)}, \dots, Z_N^{(k)})$ be a random vector indicating the chunk placement at cache k, where $Z_i^{(k)}$ is the random variable denoting the number of chunks cached from file i at cache k. Then, $P^{(k)}(\mathbf{z}) = \Pr\{\mathbf{Z}^{(k)} = \mathbf{z}\}$.

We assume that one request is generated at each time slot in the CD phase. Also, the generated request belongs to cache k with probability $p_g^{(k)}$, where $\sum_{k=1}^{K} p_g^{(k)} = 1$. It is worth noting that the request probabilities $p_g^{(k)}$ can be used to model the influence of the number of users under the coverage of different caches and their activities. On the other hand, each request is related to file i with probability p_i, regardless of its generating cache. p_i is referred to as the popularity of file i and we have $\sum_{i=1}^{N} p_i = 1$. Regarding that the popularity of the file and the requesting cache are independent, the probability that cache k requests file i is written as:

$$P(k, i) = p_i \, p_g^{(k)}. \quad (2)$$

Table 1
Notations.

Notation	Description
N	Number of files in the library
K	Number of caches
M	Cache size
ζ	Guaranteed value for Ψ
C	Number of chunks in a file
$q_i^{(k)}$	Probability that cache k stores file i
p_i	Popularity probability of file i
P_e	Error probability of ADV
Ψ	Privacy degree
Ω	Total communication cost
$p_g^{(k)}$	Request generation probability of cache k
I	Random variable (r.v.) that shows the index of the requested file
U	R.v. that shows the index of the requesting user
$\hat{i}$	Estimated file index
$\hat{k}$	Estimated cache index
Z	R.v. that shows the number of stored chunks
Y	R.v. that shows the number of non-cached chunks

When cache k queries file i, it requests for those chunks of file i that are not available in its cache. Then, the server, being aware of the chunk placements at the caches, sends the un-cached chunks of file i. As such, the server transmits $Y_i^{(k)} = C - Z_i^{(k)}$ number of chunks back to cache k over a shared link channel, where $Y_i^{(k)}$ is the random variable denoting the number of chunks transferred over the link given that cache k has requested file i. The next part describes the adversary model and its related assumptions.

2.2. Adversary model

The adversary is assumed to be a passive attacker in our system model, i.e., it does not decrypt or corrupt the communication over the shared link, e.g., due to solid encryption applied on the transferred files. The adversary in our model only eavesdrops on the communication between the server and edge caches such that it counts the number of chunks transferred in response to every request. Then, it estimates the requested file and the requesting cache based on the observed number of transferred chunks. Also, according to Kerckhoffs's principle [26], we assume that the adversary is protocol-aware, i.e., it has complete knowledge about the probabilistic caching policies, $P^{(k)}(\mathbf{z})$, the files' popularities, p_i, and the probability of generating requests by the caches, $p_g^{(k)}$, however, it has no information about the keys or the

seeds used for generating the cache placements according to JPC. This assumption corresponds to practical situations where the adversary is an authorized party in the network (honest but curious). Thus, it has full access to the network information broadcast by the server in the initiation phase, e.g., the adversary can be a code running on one of the edge-caches, which are authorized components in the network [19,27].

Although the adversary knows the probabilistic caching protocols of different caches, it does not know which specific chunks are stored at each cache. Thus, after observing the number of chunks transferred over the link, in response to one request, the adversary estimates the requested file and the requesting cache through employing the MAP rule,[1] with prior knowledge of $P^{(k)}(\mathbf{z})$, p_i, and $p_g^{(k)}$.

Let $\hat{k}(y)$ and $\hat{i}(y)$ denote the adversary's estimation of the requesting cache and requested file, respectively, given that y chunks are transferred over the shared link. Then, according to the MAP rule, $(\hat{k}(y), \hat{i}(y))$ are derived as

$$(\hat{k}(y), \hat{i}(y)) = \underset{k\in\mathcal{K}, i\in\mathcal{N}}{\operatorname{argmax}} P(k, i|Y = y) \tag{3}$$

$$= \underset{k\in\mathcal{K}, i\in\mathcal{N}}{\operatorname{argmax}} P(Y = y|k, i)p_i p_g^{(k)}, \tag{4}$$

where Y is the random variable denoting the number of chunks transferred over the shared link, and $P(k, i|y)$ is the probability that k has requested i, given that $Y = y$. Moreover, (4) is written using the Bayes' rule and (2).

2.3. Communication cost

Communication cost denoted by Ω, is defined as the average number of files transferred over the shared link in response to one request at the CD phase, i.e., $\Omega = \frac{1}{C}\mathrm{E}[Y]$. Therefore, using (2), Ω is formulated as

$$\Omega = \frac{1}{C}\sum_{k=1}^{K}\sum_{i=1}^{N} P(k, i)\mathrm{E}[Y_i^{(k)}] = \frac{1}{C}\sum_{k=1}^{K}\sum_{i=1}^{N} p_g^{(k)} p_i E[Y_i^{(k)}], \tag{5}$$

where as noted before $Y_i^{(k)}$ is the random variable denoting the number of transferred chunks, given that cache k requests file i. Moreover, $E[.]$ is the expectation operator.

2.4. Privacy degree

As discussed in Section 2.2, upon observing the number of chunks transferred over the shared link, the adversary uses MAP rule, as expressed in (3), to estimate the requesting cache and the requested file indices, i.e., $\hat{k}(y)$ and $\hat{i}(y)$, respectively. The error happens when the adversary does not detect either the requested cache or the requested file correctly. Let us define $P_{e|y}$ as the error probability of the adversary given that y chunks are observed. Then, $P_{e|y}$ is written as

$$P_{e|y} = \Pr\{U \neq \hat{k}(y) \text{ or } I \neq \hat{i}(y)|Y = y\}, \tag{6}$$

where U and I are random variables denoting the real requesting cache and requested file, respectively. Now, the privacy degree, denoted by Ψ, is defined as the error probability of the adversary and is derived as

$$\Psi = \sum_{y=0}^{C} P_{e|y} \Pr\{Y = y\}. \tag{7}$$

2.5. Example

Here, we bring a simple example in order to clarify the motivation behind using probabilistic caching for the aim of privacy-preserving. In this regard, we compare two deterministic and probabilistic caching policies in our proposed scenario with the following parameters; the library contains two files A and B with popularities $p_A = 0.8$ and $p_B = 0.2$, there is one cache with capacity one file, i.e., $k = M = 1$, and the files are not divided in smaller portions, i.e., $C = 1$. The deterministic caching scenario caches file A with probability one. According to the assumptions in Section 2.2, the adversary is aware of the caching protocol, thus, knows that file A is cached. Consequently, it infers that files A and B are requested upon observing zero and one transferred file over the shared link, respectively. This leads to an error-free detection at the adversary. On the other hand, in the probabilistic caching scenario, files A and B are cached with probabilities 0.7 and 0.3, respectively. Then, it can be seen that $\Pr(Y = 0|A)p_A = 0.7 \times 0.8 > \Pr(Y = 0|B)p_B = 0.3 \times 0.2$ and $\Pr(Y = 1|A)p_A = 0.3 \times 0.8 > \Pr(Y = 1|B)p_B = 0.7 \times 0.2$, which regarding (4), implies that the adversary always estimates file A as the requested file upon observing either zero or one file, over the shared link. Thus, the error happens when file B is requested, with probability 0.3. Therefore, through applying probabilistic caching, privacy degree increases. However, this is at the cost of increasing the communication cost. As such, in deterministic policy the communication cost Ω is equal to $\Omega = p_B = 0.2$, while in the probabilistic case, we have $\Omega = 0.3p_A + 0.7p_B = 0.35$.

Thus, when applying the probabilistic caching, a trade-off exists between the communication cost and the privacy degree. In this paper, we characterize this trade-off. In particular, in the next section, we minimize the communication cost over all probabilistic caching policies such that the privacy degree exceeds a desired threshold and show that the formulated optimization can be written as a Linear Programming (LP) optimization.

3. Problem formulation

In this section, we optimize the probabilistic caching policies at different caches in order to minimize the communication cost while keeping the privacy degree greater than a predefined threshold. Our goal is to solve the following optimization problem

$$\begin{aligned} \min_{\{P^{(k)}(\mathbf{z}):\mathbf{z}\in\mathcal{F}, k\in\mathcal{K}\}} \quad & \Omega \\ s.t. \quad & \Psi \geq \zeta, \\ & \sum_{\mathbf{z}\in\mathcal{F}} P^{(k)}(\mathbf{z}) = 1, \qquad k \in \mathcal{K}, \\ & 0 \leq P^{(k)}(\mathbf{z}) \leq 1, \qquad \mathbf{z} \in \mathcal{F},\ k \in \mathcal{K}, \end{aligned} \tag{8}$$

where Ω and Ψ are derived in (5) and (7), respectively. Also, ζ denotes the threshold associated with the privacy degree. Moreover, the second and third constraints in (8) ensures that $P^{(k)}(\mathbf{z})$ is a probability mass function. In the following, we first rewrite Ω and then Ψ in terms of the probabilistic caching policies, i.e., $P^{(k)}(\mathbf{z})$'s. In this regard, $E\left[Y_i^{(k)}\right]$ in (5) can be written as

$$E\left[Y_i^{(k)}\right] = \sum_{y=0}^{C} y \Pr\{Y_i^{(k)} = y\} = \sum_{y=0}^{C} y \Pr\{Z_i^{(k)} = C - y\}, \tag{9}$$

where in order to calculate $\Pr\{Z_i^{(k)} = C - y\}$, we need to consider any placement $\mathbf{z} \in \mathcal{F}$ that contains exactly $C - y$ chunks of i. Thus, $\Pr\{Z_i^{(k)} = C - y\}$ is written as

$$\Pr\{Z_i^{(k)} = C - y\} = \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}). \tag{10}$$

Using (9) and (10) in (5), the communication cost is derived in terms of $P^{(k)}(\mathbf{z})$'s as

$$\Omega = \frac{1}{C}\sum_{k=1}^{K}\sum_{i=1}^{N}\sum_{y=0}^{C} p_g^{(k)} p_i\, y \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}). \tag{11}$$

[1] The MAP rule is chosen since it is one of the most widely used point estimators [28].

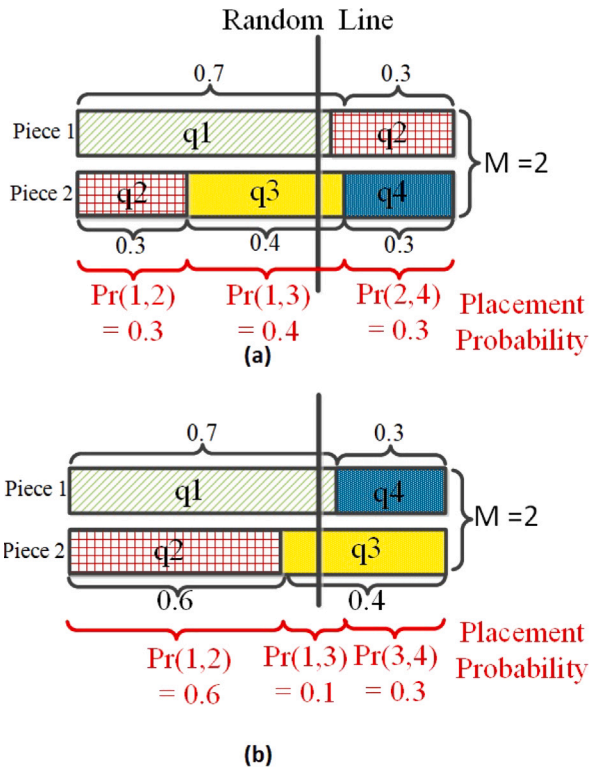


Fig. 2. Cache content placement in DPC approach, with filling order of files (a) $(1,2,3,4)$ (b) $(1,4,2,3)$.

Next we derive the privacy degree in terms of $P^{(k)}(\mathbf{z})$'s. From (6) and (7), Ψ is written as

$$\Psi = \sum_{y=0}^{C} \Pr\{U \neq \hat{k}(y) \text{ or } I \neq \hat{i}(y) | Y = y\} \Pr\{Y = y\} \tag{12a}$$

$$= \sum_{y=0}^{C} (1 - \Pr\{U = \hat{k}(y), I = \hat{i}(y) | Y = y\}) \Pr\{Y = y\} \tag{12b}$$

$$= \sum_{y=0}^{C} (1 - \max_{k,i} P(k,i|Y=y)) \Pr\{Y = y\} \tag{12c}$$

$$= 1 - \sum_{y=0}^{C} \max_{k,i} \; p_i \, p_g^{(k)} \, P(Y = y|k,i), \tag{12d}$$

where (12b) is replaced with (12c) using the MAP rule in (3). Moreover, the argument of the maximization in (12d) is written using the Bayes' rule. Using $\Pr\{Y_i^{(k)} = y\} = \Pr\{Z_i^{(k)} = C-y\}$ and (10), $P(Y = y|k,i)$ in (12d), can be written as

$$P(Y = y|k,i) = \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}). \tag{13}$$

Finally, by applying (13) in (12d), Ψ is written in terms of JPC as

$$\Psi = 1 - \sum_{y=0}^{C} \max_{k,i} \; p_i p_g^{(k)} \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}). \tag{14}$$

In the next part, we use the derived equations for Ω and Ψ to formulate the optimization problem as an LP optimization.

3.1. Linear programming optimization

Using (11) and (14) in (8), the optimization problem is rewritten as

$$\begin{aligned} \min_{\substack{\{P^{(k)}(\mathbf{z}): \\ \mathbf{z}\in\mathcal{F}, k\in\mathcal{K}\}}} \quad & \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{y=0}^{C} p_i \, p_g^{(k)} \, y \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}) \\ s.t \quad & 1 - \sum_{y=0}^{C} \max_{k,i} p_i p_g^{(k)} \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}) \geq \zeta, \\ & \sum_{\mathbf{z}\in\mathcal{F}} P^{(k)}(\mathbf{z}) = 1, \qquad k \in \mathcal{K}, \\ & 0 \leq P^{(k)}(\mathbf{z}) \leq 1, \qquad \mathbf{z} \in \mathcal{F}, \; k \in \mathcal{K}. \end{aligned} \tag{15}$$

Note that in (15), the objective function and all constraints except the first one are linear in terms of $P^{(k)}(\mathbf{z})$. In this regard, we use additional variables, Γ_y ($y \in \{0,1,\ldots,C\}$), to change the optimization problem into a linear one. In particular, the first constraint in (15) is replaced with the following inequality

$$1 - \sum_{y=0}^{C} \Gamma_y \geq \zeta. \tag{16}$$

Moreover, the following constraints on Γ_y's are added to the optimization problem

$$\Gamma_y \geq p_i \, p_g^{(k)} \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}) \quad \forall k,i,y, \tag{17}$$

where the inequalities in (17) imply that $\Gamma_y \geq \max_{k,i} \; p_i \, p_g^{(k)} \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}), \; \forall y$.

Using (16) and (17), the optimization problem turns into

$$\begin{aligned} \mathcal{P}_1 : \min_{\substack{\{P^{(k)}(\mathbf{z}): \mathbf{z}\in\mathcal{F}, k\in\mathcal{K}\}, \\ \{\Gamma_y : y\in\{0,\ldots,C\}\}}} \quad & \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{y=0}^{C} p_g^{(k)} \, p_i \, y \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}) \\ s.t. \quad & 1 - \sum_{y=0}^{C} \Gamma_y \geq \zeta \\ & \Gamma_y \geq p_i \, p_g^{(k)} \sum_{\substack{\mathbf{z}\in\mathcal{F}: \\ z_i = C-y}} P^{(k)}(\mathbf{z}), \; \forall k,i,y, \\ & \sum_{\mathbf{z}\in\mathcal{F}} P^{(k)}(\mathbf{z}) = 1, \qquad k \in \mathcal{K} \\ & 0 \leq P^{(k)}(\mathbf{z}) \leq 1, \qquad \mathbf{z} \in \mathcal{F}, \; k \in \mathcal{K}. \end{aligned} \tag{18}$$

Note that two optimization problems (15) and (18) are equivalent and thus, any optimal $P^{(k)}(\mathbf{z})$ in (18) is the solution of the optimization problem (15). As can be seen, the optimization $\mathcal{P}_1$ in (18) is linear in terms of $P^{(k)}(\mathbf{z})$'s and Γ_y's, and thus, is an LP optimization. Moreover, it is worth noting that in symmetric scenarios where caches generate the requests with almost the same probabilities, i.e., $p_g^{(k)} = p_g$, all caches have the same optimal probabilistic caching policies, and it suffices to optimize $P^{(k)}(\mathbf{z}) = P(\mathbf{z})$.

3.2. Hit ratio-constrained JPC

One advantage of chunk-based caching in JPC is to increase the cache hit ratio, where the hit ratio of a specific file at cache k is defined as the probability that at least one of its chunks exists in the cache. In practice, especially in the case of video-type contents, once the end-user is enjoying the directly received chunks from the cache, the cache requests the un-cached chunks, leading to less experienced delay at end-users. In this regard, we incorporate the average cache hit ratio constraint into the optimization problem $\mathcal{P}_1$, as in the following.

$$\sum_{k} p_g^{(k)} h^{(k)} \geq \beta, \tag{19}$$

where β is a constant threshold and $h^{(k)}$ denotes the hit-ratio of cache k. $h^{(k)}$ is written as $h^{(k)} = \sum_i p_i \sum_{\mathbf{z}\in\mathcal{F}:\ z_i\neq 0} P^{(k)}(\mathbf{z})$, where it is assumed that the request for a file hits the cache if at least one chunk of the requested file exists in the cache. It is worth noting that if we do not apply chunk-based caching in JPC, i.e., $z_i \in \{0, C\}$ instead of $z_i \in \{0, 1, 2, \dots, C\}$, then, from (11), it can be seen that the constraint $\sum_k p_g^{(k)} h^{(k)} \geq \beta$ is equivalent to $\Omega \leq 1 - \beta$.

The number of variables in the LP optimization (18) is equal to $(C+1)+KX$, where X is the size of $\mathcal{F}$, i.e., the set of all possible chunk placements at caches. A bottleneck in solving (18) is the derivation of the set of all feasible placements, which grows explosively with the number of files, chunks, and cache sizes. Although large-scale LP optimization techniques, such as CVX, can handle large-scale problems, the complexity remains in the derivation of the set $\mathcal{F}$, mainly when the edge servers have limited energy and processing power. In this regard, we propose two probabilistic caching policies with less complexity in the rest of the paper. As such, in the next section, we optimize a non-chunk-based probabilistic caching policy introduced in [24], which reduces the number of variables to NK. Then, in Section 5, we introduce the scalable version of JPC (SPC) and compare its results against the optimal JPC in Section 8.

4. Disjoint probabilistic caching (DPC)

In this section, we optimize the probabilistic caching protocol proposed in [24]. In this method, the whole file is either cached or not cached, thus, the set of feasible placements is

$$\tilde{\mathcal{F}} = \{\tilde{\mathbf{z}} | \tilde{z}_i \in \{0, C\}, i \in \{1, \dots, N\}, \sum_{i=1}^{N} \tilde{z}_i = MC\}. \tag{20}$$

Moreover, in DPC, rather than indicating a distribution over $\tilde{\mathcal{F}}$, the probability of caching each file is determined. Then, it is shown that under some condition, there exists a distribution over $\tilde{\mathbf{F}}$ that yields the chosen probabilities. In this regard, to be consistent with JPC, we refer to this policy as disjoint probabilistic caching (DPC). More precisely, DPC is defined as follows.

Definition 2. In DPC, the probability of storing file i at cache k, denoted by $\alpha_i^{(k)}$, is indicated for $\forall i \in \mathcal{N}, \forall k \in \mathcal{K}$. In order to ensure that there exists a distribution $Q^{(k)}(\tilde{\mathbf{z}})$ over $\tilde{F}$ that yields caching probabilities $\alpha_i^{(k)}$, it is required that $\sum_{i=1}^{N} \alpha_i^{(k)} = M$.

Note that in the above definition, $Q^{(k)}(\tilde{\mathbf{z}})$ yields $\alpha_i^{(k)}$ if the equality $\sum_{\tilde{\mathbf{z}}\in\tilde{\mathcal{F}}:\ \tilde{z}_i=C} Q^{(k)}(\tilde{\mathbf{z}}) = \alpha_i^{(k)}, \forall k, i$ holds. The content placement in DPC approach is based on "Madow Sampling". This strategy is proposed in [29] as a systematic sampling method, and then, is rediscovered and applied independently in the context of networked cache in [24,30]. In order to clarify this strategy, we bring a simple example here. Assume a scenario with $N = 4$ library files and one cache with capacity of two files, i.e., $K = 1$ and $M = 2$. Also, the caching probabilities of the files are chosen to be $\alpha_1^{(1)} = 0.7$, $\alpha_2^{(1)} = 0.6$, $\alpha_3^{(1)} = 0.4$, and $\alpha_4^{(1)} = 0.3$, satisfying the equality $\sum_{i=1}^{N} \alpha_i^{(1)} = 2$. The placement strategy in DPC works as follows. The cache is divided into M (2 in our example) intervals of length one, which these intervals are placed vertically under each other, as shown in Fig. 2.a. Then, the intervals are filled with $\alpha_i^{(k)}$'s (at any desired order) one after another, without replacement. Note that since $\sum_i \alpha_i^{(k)} = M$, all intervals are completely covered. Afterwards, a random number is chosen uniformly in the interval $[0, 1]$, and then, according to the chosen number, a vertical line passes through all M intervals, which indicates the set of M files to be cached. The indicated files are definitely distinct since $\alpha_i^{(k)} \leq 1$. For more detail please refer to [24]. The aforementioned example is illustrated in Fig. 2.a, where the intervals are filled with $\alpha_i^{(1)}$'s in ascending order of their indexes. Consequently, the distribution $Q^{(k)}(\tilde{\mathbf{z}})$ will be $Q^{(k)}(\tilde{\mathbf{z}} = (1, 2)) = 0.3$, $Q^{(k)}(\tilde{\mathbf{z}} = (1, 3)) = 0.4$ and $Q^{(k)}(\tilde{\mathbf{z}} = (2, 4)) = 0.3$, and zero, otherwise.

When optimizing DPC, we optimize the probability of caching individual files, i.e., $\alpha_i^{(k)}$. Then, we find a corresponding distribution over $\tilde{\mathcal{F}}$, using the DPC caching strategy explained above. In particular, the privacy-constrained optimization of communication cost under DPC is written as

$$\begin{aligned} \min_{\{\alpha_i^{(k)}: k\in\mathcal{K}, i\in\mathcal{N}\}} \quad & \Omega \\ s.t. \quad & \Psi \geq \zeta, \\ & 0 \leq \alpha_i^{(k)} \leq 1 \quad \forall k, i, \\ & \sum_{i=1}^{N} \alpha_i^{(k)} = M, \end{aligned} \tag{21}$$

where the last constraint ensures that a distribution over $\tilde{\mathcal{F}}$ can be found to satisfy the file caching probabilities $\alpha_i^{(k)}$. Moreover, the communication cost (Ω) is written in terms of $\alpha_i^{(k)}$'s as follows:

$$\Omega = 1 - \sum_{k=1}^{K} \sum_{i=1}^{N} p_g^{(k)} p_i \alpha_i^{(k)}, \tag{22}$$

where the second term in the above equation indicates the probability that no file is transferred as the response to a typical request. Also, the privacy degree is derived from (12d), except that the summation has two terms, corresponding to the values $y = 0$ and $y = C$, respectively. Thus, the privacy degree is written as:

$$\Psi = 1 - \max_{i,k} p_g^{(k)} p_i \alpha_i^{(k)} - \max_{i,k} p_g^{(k)} p_i (1 - \alpha_i^{(k)}), \tag{23}$$

where the first and second maximizations correspond to $y = 0$ and $y = C$, respectively. Using the same procedure as in Section 3.1, we can turn the optimization problem in (21) into an LP as follows

$$\begin{aligned} \mathcal{P}_2: \quad \min_{\{\alpha_i^{(k)}: k\in\mathcal{K}, i\in\mathcal{N}\}, \gamma_0, \gamma_1} \quad & \sum_{k=1}^{K} \sum_{i=1}^{N} p_g^{(k)} p_i (1 - \alpha_i^{(k)}) \\ s.t. \quad & 1 - \gamma_0 - \gamma_1 \geq \zeta, \\ & \gamma_0 \geq p_g^{(k)} p_i \alpha_i^{(k)}, \quad \forall k, i \\ & \gamma_1 \geq p_g^{(k)} p_i (1 - \alpha_i^{(k)}), \quad \forall k, i \\ & 0 \leq \alpha_i^{(k)} \leq 1, \quad \forall k, i, \\ & \sum_{i=1}^{N} \alpha_i^{(k)} = M. \end{aligned} \tag{24}$$

In the following, we compare the performance of JPC against the DPC approach and clarify its benefits over the DPC.

4.1. DPC versus JPC

In this part, we first prove the following Lemma.

Lemma 1. *The chunk-based optimization of the JPC in (18) has the same performance as the DPC optimization in (24).*

Proof. In order to prove that optimizing JPC and DPC result in the same optimum communication cost, in the first step, we prove that any feasible point of optimization $\mathcal{P}_1$ corresponds to a feasible point of optimization $\mathcal{P}_2$, with the same communication cost. Suppose that $P^{(k)}(\mathbf{z})$ and $\{\Gamma_0, \Gamma_1, \dots, \Gamma_C\}$ indicate a feasible point of $\mathcal{P}_1$ in (18). Also, let $q_{i,x}^{(k)}$ be the probability that x chunks of file i are cached at cache k in the JPC method, i.e., $q_{i,x}^{(k)} = \sum_{\mathbf{z}:\ z_i=x} P^{(k)}(\mathbf{z})$. Now we show that $\{\alpha_i^{(k)}, \gamma_0, \gamma_1\}$, defined as

$$\alpha_i^{(k)} = 1 - \frac{1}{C} \sum_{y=0}^{C} y q_{i,C-y}^{(k)}, \tag{25}$$

$$\gamma_0 = \frac{1}{C} \sum_{y=0}^{C} (C - y) \Gamma_y, \quad \gamma_1 = \frac{1}{C} \sum_{y=0}^{C} y \Gamma_y, \tag{26}$$

is a feasible solution of optimization $\mathcal{P}_2$, with the same communication cost as that of $P^{(k)}(\mathbf{z})$ in $\mathcal{P}_1$. The latter is obvious through observing that the communication costs in $\mathcal{P}_1$ and $\mathcal{P}_2$ are written as $\frac{1}{C}\sum_k \sum_i p_g^{(k)} p_i \sum_{y=0}^{C} y q_{i,C-y}^{(k)}$ and $\sum_k \sum_i p_g^{(k)} p_i (1-\alpha_i^{(k)})$, respectively, and thus, are equal according to (25). Now to show that $\{\alpha_i^{(k)}, \gamma_0, \gamma_1\}$ is a feasible solution of $\mathcal{P}_2$, note that the first inequality in $\mathcal{P}_2$ is equivalent to the first inequality in $\mathcal{P}_1$ since from (26), we have $\gamma_0+\gamma_1 = \sum_{y=0}^{C} \Gamma_y$. Moreover, the second constraint in $\mathcal{P}_1$ is rewritten in terms of $q_{i,x}^{(k)}$ as $\Gamma_y \geq p_g^{(k)} p_i q_{i,C-y}^{(k)}$. Then, through using the latter inequality in the definitions of γ_0 and γ_1 in (26), we conclude that

$$\begin{aligned} \gamma_0 &\geq p_g^{(k)} p_i \sum_{y=0}^{C} \frac{C-y}{C} q_{i,C-y}^{(k)} = p_g^{(k)} p_i \alpha_i^{(k)}, \\ \gamma_1 &\geq p_g^{(k)} p_i \sum_{y=0}^{C} \frac{1}{C} q_{i,C-y}^{(k)} = p_g^{(k)} p_i (1-\alpha_i^{(k)}). \end{aligned} \tag{27}$$

Thus, the second and third constraints in $\mathcal{P}_2$ also hold. Finally, it remains to prove the last inequality in $\mathcal{P}_2$. Since each placement $\mathbf{z} \in \mathcal{F}$ has exactly MC chunks, the average number of chunks saved in cache k under policy $P^{(k)}(\mathbf{z})$ equals MC, i.e., $\sum_{i=1}^{N} \sum_{x=0}^{C} x q_{i,x}^{(k)} = MC$. Using the latter equality and definition of $\alpha_i^{(k)}$ in (25), we conclude that $\sum_{i=1}^{N} \alpha_i^{(k)} = M$. This completes the proof of the first step.

In the second step, we need to show that any feasible solution of $\mathcal{P}_2$, $\{\alpha_i^{(k)}, \gamma_0, \gamma_1\}$, corresponds to a feasible solution of $\mathcal{P}_1$, $\{P^{(k)}(\mathbf{z}), \{\Gamma_y\}_{y=0}^{N}\}$, with the same communication cost. To show this, we define $P^{(k)}(\mathbf{z})$ as follows. We set $P^{(k)}(\mathbf{z})$ equal to zero for any placement $\mathbf{z} \in \mathcal{F}$ that caches at least one file partially, i.e., $P^{(k)}(\mathbf{z}) = 0$ if there exists i such that $z_i \notin \{0, C\}$. Moreover, using the DPC placement strategy and caching probabilities $\alpha_i^{(k)}$, we derive a distribution over $\tilde{\mathcal{F}}$ and assign it to $P^{(k)}(\mathbf{z})$. Moreover, we set $\Gamma_0 = \gamma_0$, $\Gamma_C = \gamma_1$, $\Gamma_y = 0$ for $y \notin \{0, C\}$. Then, it is observed that the constraint and objective functions in $\mathcal{P}_1$ and $\mathcal{P}_2$ are the same. In fact, the communication cost and constraints in $\mathcal{P}_1$ are written in terms of $q_{i,0}^{(k)}$ and $q_{i,C}^{(k)}$, which are equivalent to parameters $1-\alpha_i^{(k)}$ and $\alpha_i^{(k)}$, in $\mathcal{P}_2$. This completes the proof of the second step. □

The above lemma shows that caching chunks of files does not help in decreasing the communication cost while keeping the privacy degree above a specific threshold. The intuition behind the above lemma is that the privacy degree improves whenever the adversary can infer less information from the number of transferred chunks. Thus, if we cache the same number of chunks for any file in the cache, then the number of observed chunks gives no information to the adversary, given that one of the cached files is requested. Therefore, it is efficient to cache either the entire chunks of a file or none, leading to the following corollary.

Corollary 1. *The feasible set $\mathcal{F}$ in optimization (18) can be changed to $\tilde{\mathcal{F}}$. We refer to this optimization as non-chunk-based JPC.*

In fact, JPC becomes superior to DPC when cache hit ratio, as defined in Section 3.2, becomes important. Caching a few chunks of a less popular file in JPC leads to the hit-ratio of that file to be equal to one, while a hit-ratio equal to one is obtained in DPC only by caching the file entirely, which in turn increases the communication cost. Also, the feasible set $\mathcal{F}$ in JPC can be refined in order to include the desirable placements. For example, assume that we need to cache at least x chunks of a specific file(s) to satisfy its strict end-user delay constraints. Then, we can refine the feasible set $\mathcal{F}$ such that it includes only the placements that have at least x chunk of the specified file. To benefit from the advantages of JPC and cope with its complexity, in the next section, we propose a scalable version of JPC.

5. Scalable JPC

As mentioned in Section 3.2, deriving optimal JPC may be a complex and time-consuming task, especially when the number of chunks is greater than one, since it requires the computation of all feasible chunk placements. In this section, we propose a scalable version of the JPC, in which the complexity of deriving the feasible placements decreases through grouping the files into disjoint subsets. Moreover, the performance of the proposed approach can become arbitrary close to optimal JPC by increasing the number of subsets. We refer to this new version as Scalable Probabilistic Caching (SPC).

Suppose that $S = \{S_1, S_2, \ldots, S_L\}$ is a partition of $\mathcal{N}$, i.e., $\forall l,\ S_l \neq \emptyset$, $\cap_{l=1}^{N} S_l = \mathcal{N}$, and $S_l \cap S_k = \emptyset$, $\forall l, k$. Moreover, any file in S_l is more popular than any file in S_k, given that $l < k$, i.e., $p_i > p_j$, $\forall i \in S_k$, $\forall j \in S_l$. Then, assuming that the popularity of files decreases with their index, we have $S_l = \{\sum_{k=1}^{l-1} |S_k| + 1, \ldots, \sum_{k=1}^{l-1} |S_k| + |S_l|\}$. Also, let $\hat{\mathbf{z}} = (\hat{z}_1, \hat{z}_2, \ldots, \hat{z}_L)$ be a vector of length L, where $\hat{z}_l$ indicates the number of chunks cached from subset S_l. Then, the set of feasible placements, denoted by $\hat{\mathcal{F}}$, is written as

$$\hat{\mathcal{F}} = \{\hat{\mathbf{z}} = (\hat{z}_1, \ldots, \hat{z}_l) | \hat{z}_l \in \{0, 1, \ldots, |S_l|C\},\ \sum_{l=1}^{L} \hat{z}_l = MC\}. \tag{28}$$

It is worth noting that in order to choose $\hat{z}_l$ chunks from S_l, the chunks are chosen in a uniformly random manner, one after another without replacement.

Definition 3. (SPC) The scalable JPC (SPC) policy at cache k is defined as a probability distribution over $\hat{\mathcal{F}}$, denoted by $O^{(k)}(\hat{\mathbf{z}})$. Let $\hat{\mathbf{Z}}^{(k)} = (\hat{Z}_1^{(k)}, \hat{Z}_2^{(k)}, \ldots, \hat{Z}_N^{(k)})$ be a random vector indicating the chunk placement at cache k, where $\hat{Z}_i^{(k)}$ is the random variable denoting the number of chunks cached from subset i at cache k. Then, $P^{(k)}(\hat{\mathbf{z}}) = \Pr\{\hat{\mathbf{Z}}^{(k)} = \hat{\mathbf{z}}\}$.

5.1. Communication cost and privacy degree of SPC

Let $a_l = \sum_{i \in S_l} p_i$ be the probability that one of the files in subset S_l is requested. Then, the communication cost of SPC is derived as

$$\Omega = \frac{1}{C} \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{x=0}^{|S_l|C} p_g^{(k)}\ a_l\ q_{l,x}^{(k)}\ (C - \frac{x}{|S_l|}), \tag{29}$$

where $q_{l,x}^{(k)}$ denotes the probability of caching x chunks from subset S_l at cache k. Moreover, given that x chunks is selected from subset S_l, each chunk is chosen with probability $\frac{x}{|S_l|C}$, regarding the chunk selection method in SPC. Thus, the average number of un-cached chunks of any file in subset S_l is equal to $C - \frac{x}{|S_l|}$, i.e., the last multiplicative term in (29). Also, $q_{l,x}^{(k)}$ is written as

$$q_{l,x}^{(k)} = \sum_{\hat{\mathbf{z}}: \hat{z}_l = x} O^{(k)}(\hat{\mathbf{z}}). \tag{30}$$

In order to derive privacy degree, we first derive $\Pr(Y = y | i \in S_l, k)$ as

$$P(Y = y | i \in S_l, k) = \sum_{x=C-y}^{|S_l|C} q_{l,x}^{(k)} P_{x,C-y}, \tag{31}$$

where $P_{x,C-y}$ denotes the probability that $C-y$ chunks out of x selected chunks from S_l belong to the requested file i. Note that this probability is the same for any $i \in S_l$, since as mentioned before, all chunks are chosen uniformly at random. In fact, $P_{x,C-y}$ is derived as

$$P_{x,C-y} = \frac{\binom{C}{C-y}\binom{|S_l|C-C}{x-(C-y)}}{\binom{|S_l|C}{x}}, \tag{32}$$

where the nominator indicates the number of ways to choose $C-y$ chunks from the requested file and the rest $x-(C-y)$ chunks from other files. Also, the denominator shows the total number of ways of choosing x chunks from $|S_l|C$ available ones. Since $P(Y = y | i \in S_l, k)$ is independent of i, we denote it by $P(Y = y | l, k)$ hereafter, where $P(Y = y | l, k)$ denotes the probability that y chunks are transferred over

the shared link given that a file from subset S_l is requested by cache k. Using this fact and (12d), the privacy degree can be written as

$$\Psi = 1 - \sum_{y=0}^{C} \max_{l,k} p_g^{(k)} \max_{i \in S_l} \Pr(Y = y|i,k) p_i, \tag{33a}$$

$$= 1 - \sum_{y=0}^{C} \max_{l,k} p_g^{(k)} p_l^* \Pr(Y = y|l,k), \tag{33b}$$

where p_l^* indicates the popularity of the most popular file in subset S_l.

Since Ω in (29) and the arguments of maximization in Ψ in (33b) are linear functions of probability distribution $O^{(k)}(\hat{\mathbf{z}})$, we use the same procedure as in Section 3.1 to change the optimization problem of SPC to an LP optimization as follows ($y \in \{0,1,\dots,C\}, \mathbf{z} \in \hat{F}, k \in \mathcal{K}$)

$$\mathcal{P}_3 : \min_{\{\Gamma_y\},\{O^{(k)}(\hat{\mathbf{z}})\}} \frac{1}{C} \sum_{k=1}^{K} \sum_{l=1}^{L} \sum_{x=0}^{|S_l|C} p_g^{(k)} a_l q_{l,x}^{(k)} (C - \frac{x}{|S_l|}) \tag{34a}$$

$$s.t.$$

$$1 - \sum_{y=0}^{C} \Gamma_y \geq \zeta, \tag{34b}$$

$$\Gamma_y \geq p_l^* p_g^{(k)} \sum_{x=C-y}^{|S_l|C} q_{l,x}^{(k)} P_{x,C-y}, \ \forall k,l,y, \tag{34c}$$

$$\sum_{\hat{\mathbf{z}} \in \hat{F}} O^{(k)}(\hat{\mathbf{z}}) = 1, \qquad k \in \mathcal{K} \tag{34d}$$

$$0 \leq O^{(k)}(\hat{\mathbf{z}}) \leq 1, \qquad \hat{\mathbf{z}} \in \hat{F}, \ k \in \mathcal{K}, \tag{34e}$$

where $q_{l,x}^{(k)}$ is written in terms of $O^{(k)}(\hat{\mathbf{z}})$ as in (30). The above optimization can be solved through convex optimization tools, where its complexity depends on the number of subsets, i.e., L. As L increases the performance becomes closer to the optimal JPC, where at $L = N$, each subset contains exactly one file and both methods result in the same optimal caching policies. Similar to the JPC approach the following lemma is proved.

Lemma 2. *Suppose that there exists a constant value h such that for any $l \in \{1,\dots,L\}$, $1 \leq h \leq |S_l|$ and $hL \geq M$, then the performance of optimization $\mathcal{P}_3$ does not improve with C.*

Proof. Please see Appendix A. □

Examples of the above lemma are when $L \geq M$ and $h = 1$, and when $L < M$ and the sizes of all subsets are the same, i.e., $\frac{N}{L}$. In the latter case, it is enough to choose $h \geq \frac{M}{L}$, which is possible since $N > M$. Although the value of C does not affect the optimal communication cost, but hit-ratio increases as a result of increasing C since the average number of files, cached partially from each subset, increases.

The hit-ratio constraint can also be added to the optimization $\mathcal{P}_3$, as $\sum_k p_g^{(k)} h^{(k)} \geq \beta$, where $h^{(k)}$ is derived as

$$h^{(k)} = \sum_l a_l \sum_{x=0}^{|S_l|C} q_{l,x}^{(k)} \left(1 - \frac{\binom{|S_l|C-C}{x}}{\binom{|S_l|C}{x}}\right), \tag{35}$$

where last multiplicative term indicates the probability that at least one chunk from x chosen chunks of S_l is selected from the requested file.

6. Discussions

In this section, we discuss whether or not we can implement chunk-based DPC, with madow sampling content placement strategy, in order to take advantage of the chunk-based implementation as well as low complexity of DPC. In the chunk-based DPC, $\alpha_{i,c}^{(k)}$ denotes the probability of caching cth chunk of file i at cache k, where $0 \leq \alpha_{i,c}^{(k)} \leq 1$. In order to apply madow sampling for the chunk placement in this method, we divide the cache space M into MC vertically placed intervals with size $1/C$. Then, we fill the intervals with the probabilities $\beta_{i,c}^{(k)}$'s, where $\beta_{i,c}^{(k)} = \frac{\alpha_{i,c}^{(k)}}{C}$. In this case, if we choose a random cut in $[0, \frac{1}{C}]$, then the probability of choosing cth chunk of file i at cache k, will be equal to $\frac{\beta_{i,c}^{(k)}}{1/C}$, i.e., $\alpha_{i,c}^{(k)}$. Also, in order to assure that all intervals are covered completely and each chunk is chosen at most one time, we set $\sum_i \sum_c \beta_{i,c}^{(k)} = M$ and $0 \leq \beta_{i,c}^{(k)} \leq \frac{1}{C}$, respectively.

In order to write the optimization problem of the chunk-based DPC, similar to (24), we need to write Ω and Ψ in terms of $\alpha_{i,c}^{(k)}$'s. While Ω can be written as $\Omega = \frac{1}{C} \sum_{k=1}^{K} \sum_{i=1}^{N} p_g^{(k)} p_i (C - \sum_{c=1}^{C} \alpha_{i,c}^{(k)})$, where $(C - \sum_{c=1}^{C} \alpha_{i,c}^{(k)})$ is the average number of un-cached chunks of file i at cache k, writing Ψ in terms of $\alpha_{i,c}^{(k)}$ is not trivial. In fact, using (12d), we need to derive $P(Y = y|k,i)$ in terms of $\alpha_{i,c}^{(k)}$. However, in chunk-based DPC with madow sampling, $P(Y = y|k,i)$ is dependent on the filling order of the probabilities $\beta_{i,c}^{(k)}$. As an example, assume that $M = 1, C = 2$, and $\alpha_{i,1}^{(k)} = \alpha_{i,2}^{(k)} = 0.5$, and thus, $\beta_{i,1}^{(k)} = \beta_{i,2}^{(k)} = 0.25$. In one filling strategy, both probabilities $\beta_{i,1}^{(k)}$ and $\beta_{i,2}^{(k)}$ can be placed in the first rectangle of width 0.5, which results in caching exactly one chunk at any cut and thus having $P(Y = 0|k,i) = P(Y = 2|k,i) = 0$. In another filling strategy, the probabilities $\beta_{i,1}^{(k)}$ and $\beta_{i,2}^{(k)}$ are placed at the beginning of the first and second rectangles, respectively, and thus, we have $P(y = 1|k,i) = 0$. Thus, optimizing the chunk-based DPC requires optimizing the filling order of chunks in madow sampling, which is possible through exhaustive search but is not practical. Finding an efficient way to optimize the chunk-based DPC will be an interesting subject of the future research.

7. Achievable privacy degree interval

In this section, we derive the interval of achievable privacy degrees for policies JPC, DPC, and SPC. Also, we introduce the *Random Dummy Approach* (RDA) as a baseline protocol and derive its achievable privacy degrees as well. As noted before, dummy-based approaches are the baseline solutions to privacy-preserving problems [12,31]. In the *basic dummy* approach, the M most popular files are cached in each cache with probability one. However, C chunks are sent in response to every request, regardless that the requested file is cached or not. Hence, the adversary does not acquire any information from the transmitted chunks, leading to maximum privacy degree in the network. In order to provide different degrees of privacy in the dummy approach, we consider a modified version, namely, the *Random Dummy* Approach (RDA). Similar to the basic dummy approach, in RDA, the most popular files are cached in all caches. However, each request associated to a cached file is responded with C dummy chunks with probability s and zero chunks with probability $1 - s$. Moreover, the requests associated to un-cached files are responded with C chunks, as usual. Then, using (12d), at a given s, the privacy degree of RDA is calculated as $1 - p_g^{\max}\big((1-s)p_1 + \max\{p_{M+1}, sp_1\}\big)$, where $p_g^{\max} = \max_k p_g^{(k)}$, and p_{M+1} represents the popularity of $\{M+1\}$-th file, which is the most popular un-cached file. In the following, we derive the interval of achievable privacy degrees by RDA, JPC, DPC, and SPC.

RDA, JPC, and DPC: As mentioned before, in order to minimize the privacy degree, first of all, the caching policy should be deterministic so that the adversary has no ambiguity on which file has been cached. In this case, using (12d), the privacy degree is derived as $1 - p_g^{\max} p_{cached}^* - p_g^{\max} p_{uncached}^*$, where p_{cached}^* and $p_{uncached}^*$ are the popularities of the most popular cached and un-cached files, respectively. Then, in order to minimize the privacy degree, it is enough to maximize $p_{cached}^* + p_{uncached}^*$, which happens when the M most popular files are cached with probability one in every cache. This results in minimum privacy degree to be equal to $\Psi^{\min} = 1 - p_g^{\max} p_1 - p_g^{\max} p_{M+1}$. Policies DPC and JPC can choose the most popular placement with probability one, and thus, achieve $\Psi^{\min}$. To this end, it is enough to set $q_i^{(k)} = 1$ for $i \in \{1,2,\dots,M\}$ and zero otherwise in DPC, and in JPC, $P^{(k)}(\mathbf{z}) = 1$ for $\mathbf{z} = (1,2,\dots,M)$. In RDA, the most popular files are already cached, however, through

setting $s = 0$, the requests for cached files are not responded with dummy files at all, resulting in the minimum privacy degree $\Psi^{\min}$.

On the other hand, maximum privacy degree is achieved whenever the adversary cannot achieve any information from the number of chunks transferred over the shared link. In this case, the adversary chooses the most popular file as the requested file and $\text{argmax}_k\, p_g^{(k)}$ as the requesting cache, leading to a maximum privacy degree equal to $\Psi^{\max} = 1 - p_g^{\max} p_1$. $\Psi^{\max}$ is achieved in RDA and DPC through setting $s = 1$ and $q_i^{(k)} = \frac{M}{N}$, $\forall i, k$, respectively, where the latter is written considering the constraint $\sum_i q_i^{(k)} = M$ in DPC. In JPC, any probability distribution $P^{(k)}(\mathbf{z})$ that results in the same caching probability of files leads to the maximum privacy degree $\Psi^{\max}$, e.g., the probability distribution that chooses any combination M files out of N with the same probability $\frac{1}{\binom{N}{M}}$. Thus, the achievable privacy degree by JPC, DPC, and RDA is equal to $[1 - p_g^{\max}(p_1 + p_{M+1}), 1 - p_g^{\max} p_1]$.

SPC: The following lemma introduces the achievable privacy degrees by SPC.

Lemma 3. *Given subsets $\{S_1, S_2, \dots, S_L\}$, assume a placement $\hat{\mathbf{z}}$ such that $\hat{z}_i = |S_i|C$ for $i \in \{1, \dots, m-1\}$, $0 < z_m \le |S_m|C$, and $z_i = 0$ for $i > m$. In fact, $\mathbf{z}$ is the placement which chooses all chunks of the first $m-1$ subsets completely, and the remaining chunks from subset S_m such that $\sum_{l=1}^{m} \hat{z}_l = MC$. Then, the minimum and maximum privacy degree of SPC, denoted by $\Psi_{SPC}^{\min}$ and $\Psi_{SPC}^{\max}$, respectively, are derived as*

$$\Psi_{SPC}^{\min} = 1 - p_1^* p_g^{max} - \max\left\{ \frac{\binom{|S_m|C-C}{z_m}}{\binom{|S_m|C}{z_m}} p_M^* p_g^{max}, p_{M+1}^* p_g^{max} \right\} - p_M^* p_g^{max} \left(1 - \frac{\binom{|S_m|C-C}{z_m-C} + \binom{|S_m|C-C}{z_m}}{\binom{|S_m|C}{z_m}} \right), \tag{36}$$

$$\Psi_{SPC}^{\max} = 1 - p_1^* p_g^{max}.$$

Proof. Please see Appendix B. □

It is seen from the above lemma that the minimum privacy degree in SPC is dependent on C.

8. Evaluation and numerical results

In this section, we evaluate the performance of JPC, DPC and SPC approaches under different network parameters and compare their performance against RDA. Furthermore, we compare the performance of the DPC and SPC with the JPC approach without and with having constraint on average hit-ratio probability. We use the CVX toolbox of MATLAB simulator for solving LP optimizations (18), (24) and (34). Unless otherwise stated, the system parameters are considered to be $N = 12$, $K = 4$, $C = 10$ and $M = 3$. Moreover, the popularities follow Zipf distribution with parameter $\theta = 0.65$ and the request generation probabilities are $\{0.68, 0.15, 0.12, 0.05\}$.

8.1. Numerical results

In Fig. 3, the optimal communication costs of DPC and JPC approaches, derived from (18) and (24), respectively, are plotted versus the privacy degree threshold, ζ. In the case of the JPC approach, the results are plotted in the cases of $C = 1$ and $C = 10$. As can be seen, the number of chunks does not affect the optimal communication cost in the JPC approach, as proved in Lemma 1. Moreover, DPC and JPC approaches result in the same optimal performance. Also, it is observed that optimum communication cost increases with ζ. This is due to the fact that in order to provide higher privacy degree, the randomness in caching strategy increases, which leads to caching more popular files with less probability and thus, the increase of communication cost.

In Fig. 4, the communication cost of RDA, and optimal communication costs of JPC and DPC are plotted versus ζ. In order to derive

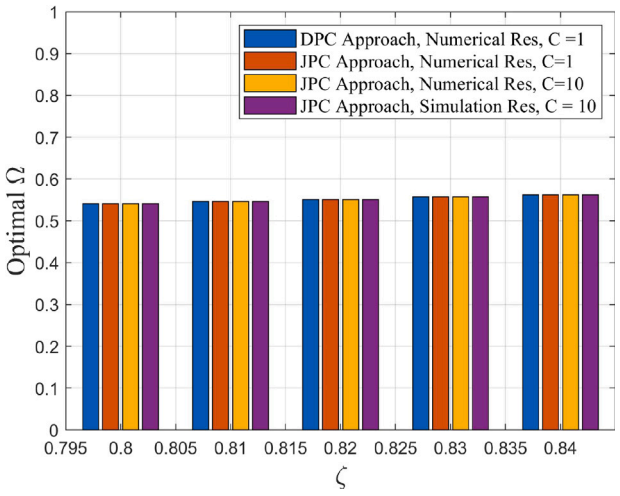


Fig. 3. Numerical and simulation results of optimal Ω versus privacy degree threshold, ζ in two approaches: DPC and JPC.

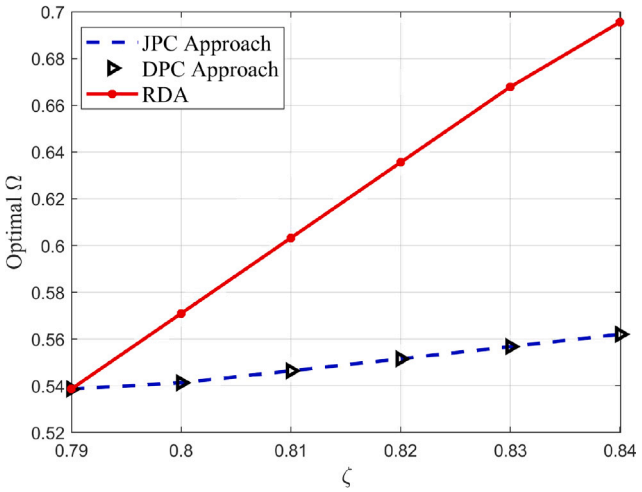


Fig. 4. Optimal Ω versus privacy degree threshold, ζ when $K = 4$, $\theta = 0.65$ and $C = 1$.

the communication cost of RDA at a given value of ζ, first we derive the parameter s of RDA approach which results in privacy degree ζ, i.e., we set $1 - p_g^{\max}\big((1-s)p_1 + \max\{p_{M+1}, s\, p_1\}\big) = \zeta$ (see Section 7). Then, the communication cost of RDA is calculated as $s\sum_{i=1}^{M} p_i + \sum_{i=M+1}^{N} p_i$. As demonstrated in Fig. 4, all three approaches have the same communication cost at the minimum privacy degree, i.e., $\zeta = 0.79$. Here, JPC and DPC cache the most popular contents, and RDA parameter s is equal to zero. However, as ζ increases, the optimum communication cost increases more rapidly in the RDA approach. This is because in RDA, in order to preserve the increased privacy degree ζ, we need to increase s, which simultaneously increases the average number of files transferred in response to all cached files. However, DPC and JPC approaches have more freedom to treat files distinctly by choosing different placements with different probabilities, thus yielding lower optimal communication cost than RDA. Moreover, according to Fig. 4, JPC (DPC) outperforms RDA, up to 19%.

Next, we investigate the impact of the number of chunks, i.e., C, on the hit-ratio-constrained JPC. In particular, when optimizing the hit-ratio-constrained JPC, we increase the value of C until there exists a feasible solution to the optimization. We refer to the minimum value of C that makes the optimization feasible as $C_{\min}$. In Fig. 5a and 5b, $C_{\min}$ and the corresponding optimal communication cost are plotted versus the hit-ratio threshold β, respectively, assuming $N = 12$ and

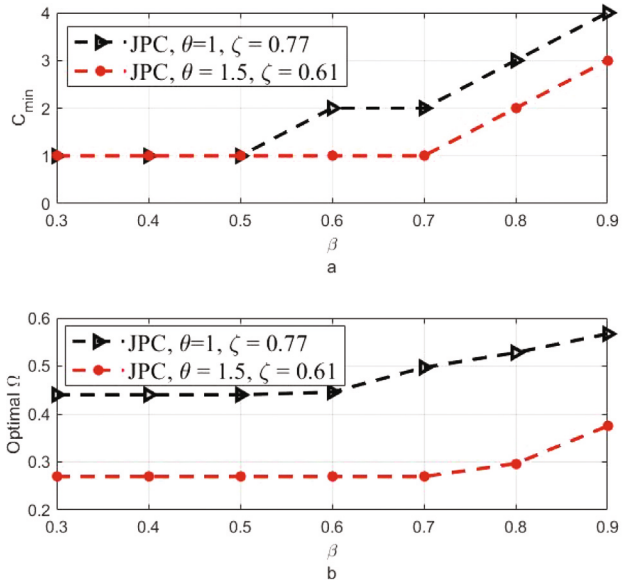


Fig. 5. (*a*) C_{min} and (*b*) optimal Ω versus hit rate threshold (β), for $K = 2$ and $\zeta = 0.61$ & 0.77.

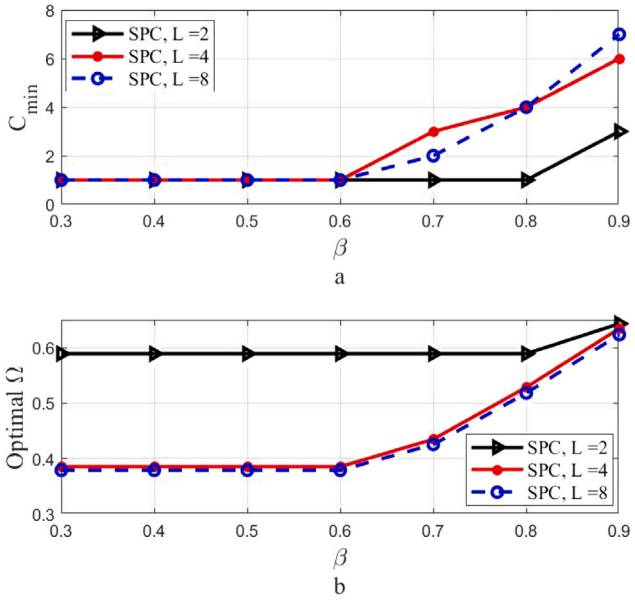


Fig. 7. (*a*) $C_{\min}$ and (b) Optimal Ω, versus hit rate threshold (β), where $K = 2$, $\zeta = 0.81$ and $\theta = 1$.

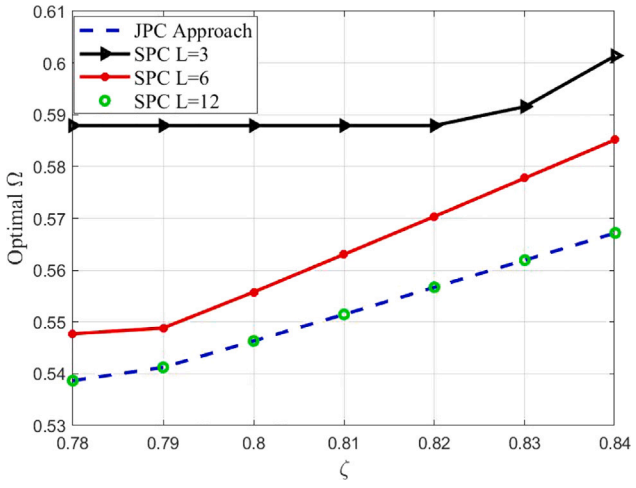


Fig. 6. Optimal Ω versus privacy degree ζ in the SPC approach, where $N = 12$, $M = 3$, $K = 2$, $\theta = 0.65$.

$M = 3$. The results are depicted for the cases that the file popularities are generated according to the Zipf distribution with parameter $\theta = 1$ and $\theta = 1.5$, and the corresponding privacy degree thresholds, ζ, are considered to be 0.77 and 0.61, respectively.

As can be observed in Fig. 5a, $C_{\min}$ increases with β since in order to increase the number of files cached partially and thus, support higher β, we need to cache smaller parts of the files, which is possible through increasing C. Meanwhile, the corresponding communication cost increases as observed in Fig. 5b, since by increasing the average hit ratio, smaller parts of the popular files are cached on average.

In Fig. 6, we compare the performance of the proposed SPC approach, at different values of L, i.e., the number of subsets, against optimal JPC, considering Zipf parameter $\theta = 0.65$, $N = 12$, and $M = 3$. Moreover, all subsets are assumed to have the same size. Also, note that the files are divided in subsets according to Section 5, i.e., the subsets are filled with files in a descending order of popularity. As can be seen in Fig. 6, the performance of optimal SPC becomes closer to the optimal JPC as L increase, where at $L = 12$, they have the same performance. Moreover, it is observed that with changing L from 12 to 3, the optimal communication cost increases at least 6% and at most 9%. However, the number of feasible placements decreases from 220 to 10, leading to less complex optimization problem. Another interesting point is that as we decrease L, the minimum achievable privacy degree increases, leading to more secure probabilistic caching strategy. As such, the minimum privacy degree increases 5% at $L = 3$ compared to $L = 12$.

In Figs. 7(a) and 7(b), $C_{\min}$, i.e., minimum value of C at which hit-ratio-constrained SPC optimization is feasible, and its corresponding optimal communication cost are plotted versus hit-ratio threshold β, respectively. As observed in Figs. 7(a), $C_{\min}$ increases with β since in order to support a higher hit-ratio, more files should be cached partially, which requires caching smaller parts of the files through increasing the value of C. Also, it is observed that when the number of subsets is small, generally $C_{\min}$ is smaller, e.g., at $L = 2$, $C_{\min}$ increases from one at hit-ratio equal to 0.8, however, at $L = 4, 8$, such a value is equal to 0.6. This is because, in the SPC approach, the chunks in each subset are chosen equally probably. Thus, the adversary gains no knowledge about the files within each subset and chooses the most popular file in each subset as the requested file. Consequently, when the number of files within each subset increases, or equivalently L becomes smaller, the error probability of the adversary, i.e., the probability that one file other than the most popular file is requested, increases. Moreover, it is observed from 7(b) that the optimal communication cost increases with β. This is because the space of the cache is dedicated more to less popular files to increase the hit-ratio, which in turn increases the communication cost.

In Fig. 8(a), the average of optimal Ω over the interval of achievable privacy degrees is plotted versus the number of subsets L. As can be seen, the average optimal Ω decreases with L since the performance of SPC becomes closer to the optimal JPC. Also, the gradient of the curves is decreasing, i.e., the difference between average optimal Ω in smaller size L, e.g., 2 and 3, is more significant than in larger sizes of L such as 8 and 10. This is important because the profit ratio is not worth the computation cost despite the minimization of optimal Ω in larger L values. As depicted in Fig. 8(b), the number of feasible placements, i.e., $|\hat{\mathcal{F}}|$ grows exponentially as L increases.

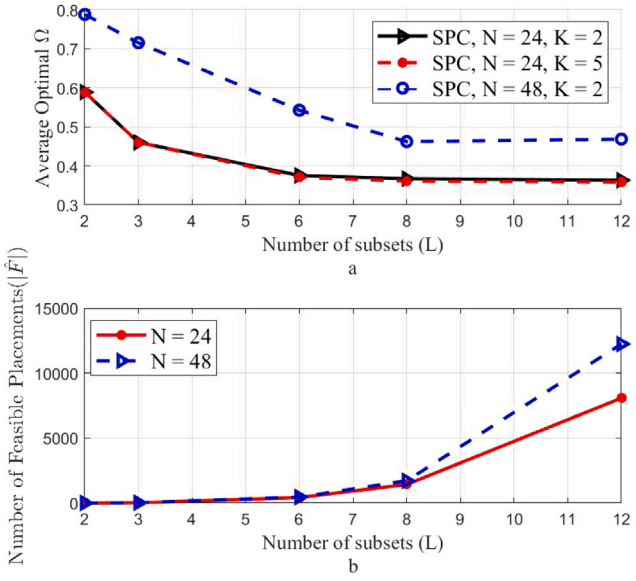


Fig. 8. (*a*) Average optimal Ω and (*b*) The number of feasible placements versus number of subsets(L) points, where $K = 2$ and $\theta = 1$.

9. Conclusion

Decreasing the communication cost in edge networks results in caching the most popular files in edge caches. However, it degrades the users' privacy since it provides the adversary with knowledge about users' interests in different files. In this paper, we proposed JPC, based on joint chunk placement at different caches, to preserve privacy in the network while minimizing the communication cost. We showed that the corresponding optimization problem is LP. However, the LP optimization requires deriving all feasible chunk placements, which is cumbersome when the system parameters are large. To overcome this, we propose a scalable-JPC approach (SPC) in which the files are grouped into small subsets, and then the placements are done based on the subsets. Numerical results revealed that the JPC approach outperforms DPC in hit-ratio-constrained scenarios and random dummy approaches in general.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Appendix A. Proof of Lemma 2

Assume that in the cases of $C = 1$ and $C > 1$, the SPC policy indicates the probability distributions $O^{(k)}(\hat{\mathbf{z}})$ and $\check{O}^{(k)}(\check{\mathbf{z}})$, respectively, where $\hat{\mathbf{z}} \in \hat{F}$ (see (28)) and $\check{\mathbf{z}} = (\check{z}_1, \check{z}_2, \dots, \check{z}_L)$ represents a feasible placement in the case of $C = 1$. In fact, $\check{z}_l$ indicates the number of files chosen from subsets S_l. Thus, $\check{\mathbf{z}} \in \check{F} = \{\check{\mathbf{z}} | \check{z}_l \in \{0, 1, \dots, |S_l|\}, 1 \le l \le L, \sum_{l=1}^{L} \check{z}_l = M\}$. In the case that $C > 1$, we refer to the optimization $\mathcal{P}_3$ in (34) as $\mathcal{P}_{3c}$, and rewrite it as:

$$\begin{aligned} \mathcal{P}_{3c}: \min_{\{\Gamma_y\}_{y=0}^{C}, O^{(k)}(\hat{\mathbf{z}})} & \sum_{k=1}^{K}\sum_{l=1}^{L} p_g^{(k)}\, a_l\, E[Y_l^{(k)}] \\ s.t. \quad & 1 - \sum_{y=0}^{C} \Gamma_y \ge \zeta, \\ & \Gamma_y \ge p_l^*\, p_g^{(k)} P(Y = y | l, k),\ \forall k, l, y, \end{aligned} \tag{37}$$

where from (34), $E[Y_l^{(k)}] = \sum_{x=0}^{|S_l|C} q_{l,x}^{(k)}(1 - \frac{x}{|S_l|C})$ is the average number of transferred files under policy $O^{(k)}(\hat{\mathbf{z}})$, given that cache k requests one file from S_l. Moreover, $P(Y = y|l, k)$ in the second constraint of (37) is derived from (31) and (32). Moreover, in the case of $C = 1$, we rewrite $\mathcal{P}_3$ as:

$$\begin{aligned} \mathcal{P}_{31}: \min_{\check{\Gamma}_0, \check{\Gamma}_1, \check{O}^{(k)}(\check{\mathbf{z}})} & \sum_{k=1}^{K}\sum_{l=1}^{L} p_g^{(k)}\, a_l\, E[\check{Y}_l^{(k)}] \\ s.t. \quad & 1 - \check{\Gamma}_0 - \check{\Gamma}_1 \ge \zeta, \\ & \check{\Gamma}_0 \ge p_l^*\, p_g^{(k)} P(\check{Y} = 0 | l, k),\ \forall k, l, \\ & \check{\Gamma}_1 \ge p_l^*\, p_g^{(k)} P(\check{Y} = 1 | l, k),\ \forall k, l, \end{aligned} \tag{38}$$

where $\check{Y}$ and $E[\check{Y}_l^{(k)}]$ are the corresponding values of Y and $E[Y_l^{(k)}]$ in the case of $C = 1$. $P(\check{Y} = y|l, k)$ and $E[\check{Y}_l^{(k)}]$ are derived similar to $P(Y = y|l, k)$ and $E[Y_l^{(k)}]$ through setting $C = 1$.

Now, let $\{o^{(k)}(\hat{\mathbf{z}}), \{\gamma_y\}_{y=0}^{C}\}$ be a feasible solution of $\mathcal{P}_{3c}$, resulting in $E[Y_l^{(k)}] = e_l^{(k)}$. Then, through the following lemmas, we show that the optimal communication cost of $\mathcal{P}_{3c}$ is less than or equal to that of $\mathcal{P}_{31}$.

Lemma 4. *If there exists a caching policy $\check{o}^{(k)}(\check{\mathbf{z}})$ over $\check{F}$, under which $E[\check{Y}_l^{(k)}] = e_l^{(k)}$, then, $\{\check{o}^{(k)}(\check{\mathbf{z}}), \check{\gamma}_0, \check{\gamma}_1\}$, with $\check{\gamma}_0$ and $\check{\gamma}_1$ defined as $\check{\gamma}_0 = \frac{1}{C}\sum_{y=0}^{C}(C - y)\gamma_y$ and $\check{\gamma}_1 = \frac{1}{C}\sum_{y=0}^{C} y\gamma_y$ is a feasible solution of $\mathcal{P}_{31}$, and results in the same communication cost as $o^{(k)}(\hat{\mathbf{z}})$ in optimization $\mathcal{P}_{3c}$.*

Proof. Regarding the definition of $\check{\gamma}_1$ and the second constraint in $\mathcal{P}_{3c}$, we conclude that $\check{\gamma}_1 \ge p_l^* p_g^{(k)} \frac{1}{C}\sum_{y=0}^{C} yP(Y = y|l, K) = p_l^* p_g^{(k)} e_l^{(k)}$ and $\check{\gamma}_0 \ge p_l^* p_g^{(k)}(1 - e_l^{(k)})$. These two inequalities are in fact the third and second constraints in $\mathcal{P}_{31}$ since on the one hand, under any policy belonging to $\check{F}$, we have $E[\check{Y}_l^{(k)}] = P(\check{Y} = 1|l, k)$, leading to $e_l^{(k)} = P(\check{Y} = 1|l, k)$. The first constraint is also satisfied since $\check{\gamma}_0 + \check{\gamma}_1 = \sum_{y=0}^{C}\gamma_y$. Finally, the communication cost under both policies $o^{(k)}(\mathbf{z})$ and $\check{o}^{(k)}(\check{\mathbf{z}})$ are the same and equal to $\sum_k \sum_l p_g^{(k)} a_l e_l^{(k)}$. □

Lemma 5. *Suppose that there exists a constant value h such that for any $l \in \{1, \dots, L\}$, $1 \le h \le |S_l|$ and $hL \ge MC$, then there exists a policy $\check{o}^{(k)}(\mathbf{z})$ over $\check{F}$ under which $E[\check{Y}_l^{(k)}] = e_l^{(k)}$.*

Proof. We introduce the policy $\check{o}^{(k)}(\mathbf{z})$ as follows. First of all, we assume that $\check{o}^{(k)}(\check{\mathbf{z}}) = 0$ for any $\check{\mathbf{z}}$ that $\exists \check{z}_i \notin \{0, h\}$, implying that either zero or h files are cached from any subset l. Also, let $q_{l,h}^{(k)}$ and $q_{l,0}^{(k)}$ denote the probabilities of caching h and zero files from subset S_l at cache k, under policy $\check{o}^{(k)}(\mathbf{z})$. Then, we have $E[\check{Y}_l^{(k)}] = q_{l,0}^{(k)} + q_{l,h}^{(k)}(1 - \frac{h}{|S_l|})$, i.e., with probabilities of $q_{l,0}^{(k)}$ and $q_{l,h}^{(k)}(1 - \frac{h}{|S_l|})$ one file is transferred in response to any file requested from S_l by cache k. Note that $1 - \frac{h}{|S_l|}$ is the probability that the requested file is not among h selected files. Then from $E[\check{Y}_l^{(k)}] = e_l^{(k)}$ and $q_{l,h}^{(k)} + q_{0,h}^{(k)} = 1$, $q_{l,h}^{(k)}$ is derived as $\frac{|S_l|(1-e_l^{(k)})}{h}$. It can be seen that $\sum_{l=1}^{L} h q_{l,h}^{(k)} = \sum_{l=1}^{L} |S_l|(1 - e_l^{(k)})$. Since $|S_l|(1 - e_l^{(k)})$ is the average number of files cached from S_l under policy $O^{(k)}(\hat{\mathbf{z}})$, we have $\sum_{l=1}^{L} |S_l|(1 - e_l^{(k)}) = M$. Consequently, $\sum_{l=1}^{L} q_{l,h}^{(k)} = \frac{M}{h}$. Now, we use the caching placement strategy of DPC, assuming that we have L files, a cache of size $\frac{M}{h}$ and caching probabilities $q_{l,h}^{(k)}$. Regarding the equality $\sum_{l=1}^{L} q_{l,h}^{(k)} = \frac{M}{h}$, a distribution over placements $\check{\mathbf{z}} \in \check{F}$ is determined, where $\frac{M}{h}$ elements of each placement is equal to h and other elements equal to zero. This distribution is assigned to $\check{o}^{(k)}(\mathbf{z})$. □

Appendix B. Proof of the privacy degree boundaries in SPC

The minimum privacy degree in the SPC is achieved when SPC tries to cache the most popular files as much as possible. Thus, the SPC chooses a placement in which all files of the subsets with lower index are chosen completely, as much as possible. Suppose that such a placement caches all files of the first $m-1$ subsets, i.e., $z_l = C|S_l|$ for $1 \le l \le m-1$, and caches only $z_m < C|S_m|$ chunks from subset $m-1$. We calculate the term $\max_{l,k} p_g^{(k)} p_l^* \Pr(Y|l,k)$ in (33b), for $y \in \{0,1,\dots,C\}$, to calculate the minimum privacy degree.

Case $y=0$: $\Pr(Y=0|l,k)$ is calculated as

$$\Pr(Y=0|l,k) = \begin{cases} 1; & l \in \{1,2,\dots,m-1\}, \\ \frac{\binom{|S_m|C-C}{z_m-C}}{\binom{|S_m|C}{z_m}}; & l=m, \\ 0; & \text{otherwise.} \end{cases} \tag{39}$$

Thus, we have

$$\begin{aligned} \max_{l,k} \; p_g^{(k)} p_l^* \Pr(Y=0|l,k) = \max \Big\{ & \max_{1\le k\le m-1} p_g^{(k)} p_l^*, \\ & p_g^{(k)} p_m^* \frac{\binom{|S_m|C-C}{z_m-C}}{\binom{|S_m|C}{z_m}}, 0 \Big\} = p_g^{\max} p_1^* \end{aligned} \tag{40}$$

Case $y=C$: $\Pr(Y=C|l,k)$ is calculated as

$$\Pr(Y=C|l,k) = \begin{cases} 0; & l \in \{1,2,\dots,m-1\}, \\ \frac{\binom{|S_m|C-C}{z_m}}{\binom{|S_m|C}{z_m}}; & l=m, \\ 1; & \text{otherwise.} \end{cases} \tag{41}$$

Thus, we have

$$\begin{aligned} \max_{l,k} \; p_g^{(k)} p_l^* \Pr(Y=C|l,k) = \max \Big\{ & 0, p_g^{\max} p_m^* \frac{\binom{|S_m|C-C}{z_m}}{\binom{|S_m|C}{z_m}}, \\ & \max_{m+1\le l\le L} p_g^{(k)} p_l^* \Big\} = \max\Big\{ p_g^{\max} p_m^* \frac{\binom{|S_m|C-C}{z_m}}{\binom{|S_m|C}{z_m}}, p_g^{\max} p_{m+1}^* \Big\}. \end{aligned} \tag{42}$$

Case $y \in \{1,\dots,C-1\}$: In this case, $\Pr(Y=y|l,k)=0$ for $l \ne m$ since any file requested from $m-1$ first subsets and from subsets $m+1$ to L leads to 0 and C chunks transferred, respectively. Thus, for $y \ne 0, C$, we have $\max_{l,k} \; p_g^{(k)} p_l^* \Pr(Y=y|l,k) = p_g^{\max} p_m^* \Pr(Y=y|m,k)$. Consequently, we have

$$\begin{aligned} & \sum_{y=1}^{C-1} \max_{l,k} \; p_g^{(k)} p_l^* \Pr(Y=y|l,k) \\ & = p_g^{\max} p_m^* (1 - \Pr(Y=0|m,k) - \Pr(Y=C|m,k)) \\ & = p_g^{\max} p_m^* \Big(1 - \frac{\binom{|S_m|C-C}{z_m}}{\binom{|S_m|C}{z_m}} - \frac{\binom{|S_m|C-C}{z_m-C}}{\binom{|S_m|C}{z_m}}\Big). \end{aligned} \tag{43}$$

From (40), (42), (43), and (33b), $\Psi_{\text{SPC}}^{\min}$ in (36) can be concluded. Moreover, the maximum privacy degree in SPC is achieved when all possible placements are chosen with the same probability. In this case, the adversary chooses the most popular file as the requested file and $\text{argmax}_k \; p_g^{(k)}$ as the requesting users, thus, leading to a maximum privacy degree equal to $1 - p_1^* p_g^{\max}$.

References

[1] X. Xiong, K. Zheng, L. Lei, L. Hou, Resource allocation based on deep reinforcement learning in IoT edge computing, IEEE J. Sel. Areas Commun. 38 (6) (2020) 1133–1146.

[2] S. Liu, C. Zheng, Y. Huang, T.Q. Quek, Distributed reinforcement learning for privacy-preserving dynamic edge caching, IEEE J. Sel. Areas Commun. (2022).

[3] A. Yousefpour, C. Fung, T. Nguyen, K. Kadiyala, F. Jalali, A. Niakanlahiji, J. Kong, J.P. Jue, All one needs to know about fog computing and related edge computing paradigms: A complete survey, J. Syst. Archit. (2019).

[4] W.Z. Khan, E. Ahmed, S. Hakak, I. Yaqoob, A. Ahmed, Edge computing: A survey, Future Gener. Comput. Syst. 97 (2019) 219–235.

[5] P. Ranaweera, A.D. Jurcut, M. Liyanage, Survey on multi-access edge computing security and privacy, IEEE Commun. Surv. Tutor. 23 (2) (2021) 1078–1124.

[6] J. Ni, K. Zhang, A.V. Vasilakos, Security and privacy for mobile edge caching: Challenges and solutions, IEEE Wirel. Commun. 28 (3) (2020) 77–83.

[7] L. Xiao, X. Wan, C. Dai, X. Du, X. Chen, M. Guizani, Security in mobile edge caching with reinforcement learning, IEEE Wirel. Commun. 25 (3) (2018) 116–122.

[8] S. Cui, M.R. Asghar, G. Russello, Multi-cdn: Towards privacy in content delivery networks, IEEE Trans. Dependable Secure Comput. (2018).

[9] X. He, R. Jin, H. Dai, Physical-layer assisted privacy-preserving offloading in mobile-edge computing, in: ICC 2019-2019 IEEE International Conference on Communications, ICC, IEEE, 2019, pp. 1–6.

[10] H. Ko, H. Lee, T. Kim, S. Pack, LPGA: Location privacy-guaranteed offloading algorithm in cache-enabled edge clouds, IEEE Trans. Cloud Comput. (2020).

[11] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, A. Iyengar, Location privacy-preserving mechanisms in location-based services: A comprehensive survey, ACM Comput. Surv. 54 (1) (2021) 1–36.

[12] B. Niu, Z. Zhang, X. Li, H. Li, Privacy-area aware dummy generation algorithms for location-based services, in: 2014 IEEE International Conference on Communications, ICC, IEEE, 2014, pp. 957–962.

[13] K.P. Coopamootoo, Usage patterns of privacy-enhancing technologies, in: 2020 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2020, pp. 1371–1390.

[14] D. Andreoletti, O. Ayoub, S. Giordano, G. Verticale, M. Tornatore, Privacy-preserving caching in ISP networks, in: 2019 IEEE 20th International Conference on High Performance Switching and Routing, HPSR, IEEE, 2019, pp. 1–6.

[15] D. Andreoletti, O. Ayoub, C. Rottondi, S. Giordano, G. Verticale, M. Tornatore, A privacy-preserving protocol for network-neutral caching in ISP networks, IEEE Access 7 (2019) 160227–160240.

[16] G. Acs, M. Conti, P. Gasti, C. Ghali, G. Tsudik, C.A. Wood, Privacy-aware caching in information-centric networking, IEEE Trans. Dependable Secure Comput. 16 (2) (2017) 313–328.

[17] J. Cui, L. Wei, H. Zhong, J. Zhang, Y. Xu, L. Liu, Edge computing in VANETs-An efficient and privacy-preserving cooperative downloading scheme, IEEE J. Sel. Areas Commun. 38 (6) (2020) 1191–1204.

[18] R. Schlegel, S. Kumar, E. Rosnes, A.G. i Amat, Privacy-preserving coded mobile edge computing for low-latency distributed inference, IEEE J. Sel. Areas Commun. (2022).

[19] S.B. Hassanpour, A. Diyanat, A. Khonsari, S.P. Shariatpanahi, A. Dadlani, Context-aware privacy preservation in network caching: An information theoretic approach, IEEE Commun. Lett. 25 (1) (2020) 54–58.

[20] Z. Yu, J. Hu, G. Min, Z. Wang, W. Miao, S. Li, Privacy-preserving federated deep learning for cooperative hierarchical caching in fog computing, IEEE Internet Things J. (2021).

[21] Z. Yu, J. Hu, G. Min, Z. Zhao, W. Miao, M.S. Hossain, Mobility-aware proactive edge caching for connected vehicles using federated learning, IEEE Trans. Intell. Transp. Syst. 22 (8) (2020) 5341–5351.

[22] F. Shi, L. Fan, X. Liu, Z. Na, Y. Liu, Probabilistic caching placement in the presence of multiple eavesdroppers, Wirel. Commun. Mob. Comput. 2018 (2018).

[23] B. Niu, Q. Li, X. Zhu, G. Cao, H. Li, Enhancing privacy through caching in location-based services, in: 2015 IEEE Conference on Computer Communications, INFOCOM, IEEE, 2015, pp. 1017–1025.

[24] B. Blaszczyszyn, A. Giovanidis, Optimal geographic caching in cellular networks, in: 2015 IEEE International Conference on Communications, ICC, IEEE, 2015, pp. 3358–3363.

[25] X. Lin, J. Xia, Z. Wang, Probabilistic caching placement in UAV-assisted heterogeneous wireless networks, Phys. Commun. 33 (2019) 54–61.

[26] H.C. Van Tilborg, S. Jajodia, Encyclopedia of Cryptography and Security, Springer Science & Business Media, 2014.

[27] R. Roman, J. Lopez, M. Mambo, Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges, Future Gener. Comput. Syst. 78 (2018) 680–698.

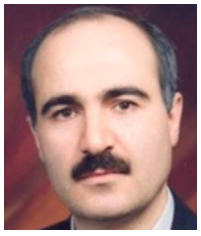
[28] J.Y. Koh, D. Leong, G.W. Peters, I. Nevat, W.-C. Wong, Optimal privacy-preserving probabilistic routing for wireless networks, IEEE Trans. Inf. Forensics Secur. 12 (9) (2017) 2105–2114.

[29] W.G. Madow, L.H. Madow, On the theory of systematic sampling, I, Ann. Math. Stat. 15 (1) (1944) 1–24.

[30] S. Ioannidis, E. Yeh, Adaptive caching networks with optimality guarantees, ACM SIGMETRICS Perform. Eval. Rev. 44 (1) (2016) 113–124.

[31] H. Lu, C.S. Jensen, M.L. Yiu, Pad: ?privacy-area aware, dummy-based location privacy in mobile services, in: Proceedings of the Seventh ACM International Workshop on Data Engineering for Wireless and Mobile Access, ACM, 2008, pp. 16–23.

Seyedeh Bahereh Hassanpour received the B.Sc. and M. Sc. degrees in computer engineering from Shahid Beheshti University (SBU) and Khajeh Nasir Toosi University (KNTU), Tehran, Iran, in 2012 and 2016, respectively. She is currently pursuing a Ph.D. degree from the Department of Electrical and Computer Engineering, University of Tehran, Iran. Her research interests include Cellular Networks, Edge Networks, Network Optimization, content placement in distributed caches.



Ahmad Khonsari received the B.Sc. degree in electrical and computer engineering from Shahid Beheshti University, Iran, and M.Sc. degree in computer engineering from the Iran University of Science and Technology (IUST), Iran, and a Ph.D. degree in computer science from the University of Glasgow, U.K., in 1991, 1996, and 2003, respectively. He is currently an Associate Professor with the Department of Electrical and Computer Engineering, University of Tehran, Iran, and a Researcher with the School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Iran. His research interests include simulation and data analysis, performance modeling/evaluation, wired/wireless networks, cloud, and distributed systems, and high-performance computer architectures.

Masoumeh Moradian received the B.S., M.S., and Ph.D. degrees from the Sharif University of Technology, Tehran, Iran, in 2007, 2010, and 2016, respectively, all in electrical engineering. She was a Visiting Scholar with the Chinese University of Hong Kong in 2015. She is currently a Postdoctoral Researcher with the School of Computer Science, Institute for Research in Fundamental Sciences (IPM), Tehran, Iran. Her current research interests include energy harvesting communication networks, queueing theory, and network stochastic optimization.

Seyed Pooya Shariatpanahi received the B.Sc., M.Sc., and Ph.D. degrees from the Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran, in 2006, 2008, and 2013, respectively. He is currently an Assistant Professor with the School of Electrical and Computer Engineering, University of Tehran. Before joining the University of Tehran, he was a Researcher with the Institute for Research in Fundamental Sciences (IPM), Tehran. His research interests include information theory, network science, wireless communications, and complex systems. He was a recipient of the Gold Medal at the National Physics Olympiad, in 2001.