Optimizing Retention-Aware Caching in Vehicular Networks

Tao Deng, *Student Member, IEEE*, Pingzhi Fan, *Fellow, IEEE*, and Di Yuan, *Senior Member, IEEE*

***Abstract*— Caching is an effective way to address the challenges due to explosive data traffic growth and massive device connectivity in fifth-generation (5G) networks. Currently, few works on caching pay attention to the impact of the time duration for which content is stored, called retention time, on caching optimization. The research on retention time is motivated by two practical issues, i.e., flash memory damage and storage rental cost in cloud networks, together giving rise to the storage cost. How to optimize caching contents taking the storage cost into consideration is a challenging problem, especially for the scenarios with cache-enabled mobile nodes. In this paper, a retention-aware caching problem (RACP) in vehicular networks is formulated, considering the impact of the storage cost. The problem's complexity analysis is provided. For symmetric cases, an optimal dynamic programming (DP) algorithm with polynomial time complexity is derived. For general cases, a low complexity and effective retention aware multi-helper caching algorithm (RAMA) is proposed. Numerical results are used to verify the effectiveness of the algorithms.**

***Index Terms*— Caching, storage cost, vehicular networks.**

I. Introduction

A. Motivation

In fifth-generation (5G) networks, caching provides significant gains in terms of reducing the backhaul burden and the content downloaded delay, etc., via storing the required content at the edge devices [1]–[3]. Recently, there is a new research line attracting interest on caching, referred to as retention-aware caching [4]–[6], considering the impact of storage cost on optimal caching. Here, retention time denotes a time duration that a piece of data is written in a memory. Research taking storage cost into consideration is inspired by two practical issues, i.e., storage rental cost and memory usage cost [4], [7]. A longer retention time will result in a higher rental cost and memory usage cost, thus leading to a higher storage cost.

In mobility scenarios, e.g., vehicular networks, if the vehicles store contents, they can exchange their interesting contents when they move into the communication range of each other, thus helping the networks for offloading. In this scenario, the storage capacity is smaller than that of a usual base station. Due to the selfish behaviors or privacy concerns of vehicles, they tend to prefer to use limited caching capacity to serve themselves. If a vehicle contributes a part of storage to help the networks for offloading, the caching capacity to serve the vehicle itself will be reduced, thus bringing negative effect to itself. In addition, the vehicle needs to consume its energy in order to transmit data to another vehicle. Finally, the memory lifetime of the vehicle will be shortened due to caching contents needed by other vehicles. The above three aspects result in the storage cost in vehicular networks. In general, the longer the retention time of contents, the higher the storage cost. On the other hand, from the network perspective, using a long retention time increases the probability of obtaining the requested contents from edge devices, thus decreasing the downloading cost from the network server. Therefore, there is a clear tradeoff between the storage cost and the downloading cost. In [7], the storage cost is modeled as an increasing convex polynomial function with respect to retention time. This is because that, first, memory usage cost is directly proportional to the retention time, justifying the choice of an increasing function. Second, with the aging of memory damage, the caused damage grows for dealing with the same amount of workload over time, justifying the choice of a convex function. Finally, the storage cost function is a non-linear function with respect to retention time, justifying the choice of a polynomial function. By adjusting the coefficients of the polynomial function, a variety of functions can be obtained.

For caching at mobile nodes, making the best of mobility information can significantly improve caching efficiency [8]–[15]. The work in [9] modeled a maximum data offloading ratio problem incorporating mobility. The work in [10] studied the impact of mobility on caching performance by taking into account the communication time between users. The work in [11] investigated the impact of the selfishness of users. The study in [12] modeled a cost optimal caching problem. Based on [12], the work in [13] further derived an approximation of the cost optimal caching problem achieving linearization. The study in [14] addressed a delay optimal caching problem with guarantee on expected network load ratio. In [15], a maximum cache hit ratio problem was introduced, and a green and mobility-aware caching model was presented. Comparing our work to works [9]–[15], the differences are as follows. First, our work investigates retention aware caching, which is not present in these works. Second, our work considers that the communication time duration of one contact is a variable. Thus, the amount of data exchanged of one contact is variable too, while it is assumed to be

Manuscript received October 8, 2018; revised March 13, 2019 and May 22, 2019; accepted May 30, 2019. Date of publication June 6, 2019; date of current version September 16, 2019. The work of T. Deng and P. Fan is supported by the National Science Foundation of China (NSFC, No. 61731017), and the 111 Project (No. 111-2-14). The work of D. Yuan has been funded by the Swedish Research Council Grant 2018-05247. The associate editor coordinating the review of this paper and approving it for publication was V. Stankovic. *(Corresponding author: Tao Deng.)*

T. Deng and P. Fan are with the School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China (e-mail: dengtaoswjtu@foxmail.com; p.fan@ieee.org).

D. Yuan is with the Department of Information Technology, Uppsala University, 751 05 Uppsala, Sweden (e-mail: di.yuan@it.uu.se).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCOMM.2019.2921368

constant in [9], [12]–[14]. Finally, although [10] considered the variation in the communication time duration, it focused on modeling the performance metric, rather than solving optimization problems. The work in [16] investigated the impact of the storage cost on caching optimization in vehicular networks. However, the system model in [16] assumed that a complete content can be transmitted during one contact, thus ignoring the fact that the contact duration is short. Our model relaxes this assumption and addresses a more general scenario.

B. Our Contribution

There is no study that explores the impact of the storage cost on caching optimization in mobility scenarios. The purpose of our paper is to deal with this aspect. More specifically, we investigate retention-aware caching in vehicular networks. Vehicular networks are representative mobility scenarios and have been regarded as one of the main application scenarios in 5G [17]–[20]. Nowadays, the on-board users during their journey want to enjoy entertainment applications, e.g., videos. A reasonable content delivery scheme can help the networks avoid repeated transmission of contents. Thus, it is of interest to investigate content deliveries in vehicular networks. In addition, the network operator have provided dedicated bandwidth for vehicle-to-vehicle (V2V) communications. Finally, the mobility traces of the real-world vehicles follow some specific distributions, e.g., exponential distribution, which should be taken into account for modeling and analysis for vehicular scenarios. More specifically, we consider that some vehicles, referred to as *requesters*, are interested in requesting some contents. Other vehicles, referred to as *helpers*, are willing to help the network offload requests from requesters by caching contents. When a requester moves into the communication range of a helper, called one contact, the former can collect the desired data from the latter. The contributions of our work are included as follows.

- First, a retention-aware caching problem (RACP) in vehicular networks is modeled, featuring two aspects arising in realistic scenarios. 1) The contents are different in terms of size. 2) We address the variable communication time duration of one contact, unlike most of the works on opportunistic vehicular communications.
- Second, the hardness of RACP, in terms of the computational complexity, is proved based on a reduction from the 3-satisfiability (3-SAT) problem that is NP-complete [21].
- An optimal dynamic programming (DP) algorithm is derived to achieve RACP's global optimum for symmetric scenarios where the helpers' caching capacities as well as the contact rates are uniform. We remark that for this case with uniformity, the problem remains combinatorial, and a native formulation gives a non-linear integer programming model. Yet we show that this problem setup is tractable.
- To tackle general system scenarios, a low-complexity retention aware multi-helper caching algorithm (RAMA) is proposed. The algorithm is based on a key observation that the optimal caching solution of a helper can be computed, with respect to the current solutions of other helpers.
- Finally, we evaluate the performance of the DP and RAMA algorithms via comparing them to the popularity-based caching algorithm (PCA) and the random caching algorithm (RCA). In symmetric scenarios, DP serves as a benchmark to verify the performance of RAMA. The results show that for RAMA, the gap to the global optimum is less than 2%, demonstrating that RAMA achieves excellent near-optimal performance. Additionally, RAMA outperforms PCA and RCA.

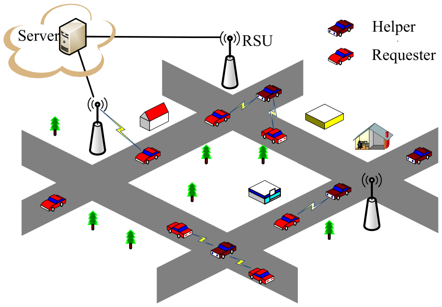

Fig. 1. System scenario.

C. Existing Studies

There are a few works that investigated retention-aware caching. In [4], the authors considered a hierarchical network topology consisting of a server, cache-enabled nodes, and users. In this topology, each user communicates with one cache-enabled node. A proactive retention-aware caching problem (PRAC) was formulated. In addition, two types of storage cost, i.e., linear storage cost and convex storage cost, were considered. The investigations in [5], [6] further extended the model in [4]. In [5], [6], each user can associate with multiple nodes, and the number of users associated with any node has an upper bound. In performance evaluation, they considered that 5% of users change their positions for each period. In our work, we extend this setup and consider a more general mobility scenario where the users' positions may be changed at any time.

The remainder of this work is organized as follows. In Section II, we present the system model, cost model, problem formulation, and complexity analysis. In Section IV, we derive the DP algorithm. Section V develops the RAMA algorithm. In Section VI, we introduce performance evaluation. Finally, we conclude this paper in Section VII.

II. System Model and Problem Formulation

A. System Model

We investigate a vehicular network scenario which consists of a content server having all the contents, $R + H$ vehicles, and road side units (RSUs) providing signal coverage for vehicles, as shown in Fig. 1. Denote by $\mathcal{R}$ the set of vehicles that are interested in requesting some contents, referred to as *requesters*. Denote by $\mathcal{R} = \{1, 2, \ldots, R\}$ the index of $\mathcal{R}$. Denote by $\mathcal{H}$ the set of vehicles that are willing to help the network offload requests from the requesters, referred to as *helpers*. Denote by $\mathcal{H} = \{1, 2, \ldots, H\}$ the index of $\mathcal{H}$. Helper k, $k \in \mathcal{H}$, is equipped with a cache of size s_k. Note that caching at both RSUs and helpers results in a hierarchical caching architecture. The modeling for this architecture is quite complex. To derive a feasible analysis model, in our paper we consider only caching at helpers. There are a total of J contents. Denote by $\mathcal{J} = \{1, 2, \ldots, J\}$ the index of $\mathcal{J}$. For content j, $j \in \mathcal{J}$, its size is l_j. In addition, for any helper, a content is either fully stored or not stored at all.

In vehicular networks, some of the vehicles are interested in requesting some contents in a specific time duration, and the others are not. The vehicles having no content requests report their current storage capacity information to the network operator. It is up to the operator to select (e.g., randomly) which vehicles to be helpers. The operator will provide some benefits to the vehicles with additional storage capacity by some incentive schemes [22], [23]. A detailed analysis of incentive schemes is beyond the scope of this paper, however. Note that even though the vehicles have spare storage capacities, some of them may not be willing to become helpers to help the operator offload data due to selfishness or privacy. In addition, from the perspective of the operator, even though some vehicles want to become helpers, the operator chooses only a limited number of vehicles as helpers. This is because the operator needs to offer some benefits to helpers. All the vehicles periodically broadcast cooperative awareness messages (CAM). The CAM includes the vehicles' speed, location, and direction information. The content transmission process from the helpers to the requesters is divided into two phases: the discovery phase and the communication phase [24], [25]. In the discovery phase, a helper uses its CAM to detect the neighboring requesters within the communication range. If a requester wants to establish a direct communication with the helper, the requester will send an acknowledgement and its CAM information to the helper. After this, the discovery phase is completed. Then, the helper and the requester can communicate with each other. Currently, the network operator has provided dedicated and sufficient resources for vehicle-to-vehicle (V2V) communications. For example, 802.11p (IEEE standard for wireless access in vehicular environments, WAVE) working in 5.86-5.925 GHz is used for dedicated short range communication (DSRC) [26].

Communication between requesters and helpers occurs when they move into the communication range of each other [27]–[29], called a contact. Our model assumes that the contact between requester i and helper k follows a Poisson distribution with rate λ_{ik}. The assumption is based on the following two facts. First, some studies have suggested a Poisson distribution to characterize the contact process of vehicles, e.g., [27], [30]. Moreover, by analyzing the mobility traces of the real-world vehicles, an exponential distribution can characterize the tail behavior of the inter-contact time distribution [31], demonstrating that our is reasonable. In vehicular networks, due to the movement of vehicles, the communication time duration of any two vehicles is short, resulting in that it may be infeasible to transmit a complete content in a limitation duration. Thus, the effect of communication time duration cannot be ignored. There are some works that investigated the characterization of communication time durations in real-world vehicular scenarios. For example, the work in [32] investigated this aspect via adopting the realistic vehicle mobility traces in [31]. The experiment results manifested that an exponential distribution can describe at least 80% of the communication time duration distributions. Based on this fact, our system model assumes that the communication time duration of one contact follows an exponential distribution with parameter μ. Same as [9], [10], [15], [27], we assume that the inter-contact and communication times are independent.

In vehicular networks, due to the intermittent connection of vehicles, the recovery of any content requested can tolerate a delay duration. We assume that all the contents have the same delay tolerant duration. This assumption is common [4]–[6], [9], [12]–[14]. We consider a slotted time system in which a time period includes T time slots[1] with equal length θ. Here, the time duration of a slot is equal to the delay tolerance duration. For any content, if it is written to a cache and stored there for some amount of retention time (i.e., how many consecutive time slots), it will be erased after this time duration. Thus, caching is not independent across slots. Their relationship is determined by the retention time. In addition, due to the limitation of cache capacity, in order to maximize the system performance, it is vital to optimize which contents to cache and for how many consecutive time slots to cache. A caching decision is described by $\boldsymbol{x}$ which is an $H \times J$ retention time matrix of the H helpers for the J contents. In the matrix, the $(k, j)^{th}$ entry, i.e., x_{kj}, denotes the retention time of content j at helper k, $x_{kj} \in \{0, 1, 2, \ldots, T\}$, where $x_{kj} = 0$ represents the case that helper k does not store content j. The retention time of each content at each helper is determined by caching optimization that takes place at the beginning of the first time slot under consideration.

In our system scenario, all the requesters are active. Namely, all the requesters are interested in requesting some contents in each slot. Same as [4]–[6], we assume that each requester will request a content that is independent of the specific time slot. Denote by w_{ij} the probability that requester i requests content j, $\sum_{j \in \mathcal{J}} w_{ij} = 1$. Note that our system model can be extended to include idle requesters via introducing a dummy content. For the dummy content, the amount of data to recover is zero. Hence, $\sum_{j \in \mathcal{J}} w_{ij} = 1$ is more general than it appears. In the content recovery model, a requester collects the data of the requested content from the encountered helpers with the presence of a delay tolerance. The order of magnitude of the delay tolerance is at least minute. If we consider the instantaneous data transmission rate with interference at at a micro time scale, the modeling aspect becomes quickly very complex. To derive a tractable model, in our paper we consider the average data transmission rate. Denote by v the mean data transmission rate during communications.

B. Cost Model

Fig. 2 shows the content recovery model. In this figure, when requester i requests content j in time slot t, it will

[1] Note that, the time duration of a slot is different from that in LTE. Here, the order of magnitude of slot is minute at least.

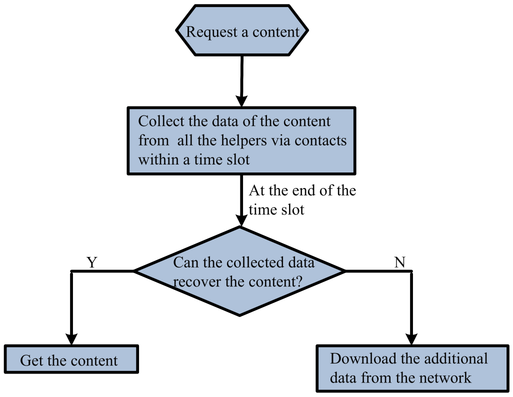
Request a content
Collect the data of the content from all the helpers via contacts within a time slot
At the end of the time slot
Can the collected data recover the content?
Y
N
Get the content
Download the additional data from the network

Fig. 2. Content recovery model.

first try to collect the data of content j from the encountered helpers. If the amount of collected data of j is less than l_j at the end of the time slot, requester i has to download the rest of data from the server in order to recover j. Denote by D_{ijt} the expected amount of collected data of j by i from the encountered helpers in t. At any time instant, a requester can only collect data from a helper. If a requester contacts multiple helpers with the requested content at the same time, it will randomly choose a helper to collect. Denote by N_{ij} the total number of contacts that requester i has with the helpers having j in t. N_{ij} is a random variable following a Poisson distribution with mean n_{ijt}, $n_{ijt} = \sum_{k\in\mathcal{H}} 1_{\{x_{kj}\geq t\}}\lambda_{ik}\theta$. In the expression of n_{ijt}, $1_{\{x_{kj}\geq t\}}$ is one if and only if helper k stores content j in t, otherwise it is zero. Therefore, D_{ijt} can be expressed as

$$D_{ijt} = \mathbb{E}(\min(\sum_{n=1}^{N_{ij}} d_n v, l_j)), \tag{1}$$

where d_n represents the communication time duration of the nth contact and $\mathbb{E}(\cdot)$ represents the expectation operator. By our assumption, d_n follows an exponential distribution, thus leading to that $\sum_{n=1}^{N_{ij}} d_n$ is a compound Poisson process. It is well known that for a compound Poisson process, it is hard to derive its close form expression. However, in general, the communication time duration is short in vehicular networks due to high mobility. Therefore, a normal distribution, i.e., $\mathrm{N}(\frac{n_{ijt}}{\mu}, 2n_{ijt}\mu^2)$, can approximate the distribution of $\sum_{n=1}^{N_{ij}} d_n$ when μ is relatively large. This fact has been proved in Lemma 1 of [27]. With this approximation, the expression of D_{ijt} can be obtained in (2), as shown at the bottom of this page, where $\phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}}\mathrm{e}^{-\frac{y^2}{2}}dy$. If $D_{ijt} < l_j$ at the end of t, requester i has to download $l_j - D_{ijt}$ amount of data from the server. As the time duration for downloading data from the network is far less than the delay tolerance duration of contents, here we neglect the effect of the former (In general, the order of magnitude of the former is ms, and the order of magnitude of the latter is minute). Denote by P_{ijt} the network load probability which is the proportion of data that requester i has to download from the server in order to recover content j at the end of slot t, and $P_{ijt} = \frac{l_j - D_{ijt}}{l_j}$. Same as [4]–[6], we focus on that the downloading cost is due to the downloading operation from the network, and that cost incurred with respect to the encountered helpers is negligible. At the end of a time slot, if the collected date cannot recover the requested content, the network server will be used due to the incomplete collection during the time slot. In the next time slot, the requester may request another content. The downloading process from the server is independent of the process of requesting a new content at the next slot [4]–[6]

Caching contents in helpers results in storage cost. Denote by $f(x)$ the storage cost of storing a content of unit size for retention time x. Referring to [7], we assume that $f(x)$ is modeled as an increasing convex polynomial function of degree m with respect to x, given by $f(x) = a_m x^m + a_{m-1}x^{m-1} + \cdots + a_1 x + a_0$ and $f(0) = 0$. For example, if $f(x) = x^2$, the storage cost for 1 slot and 2 slots are 1 and 4, respectively. Denote by α the weight of the storage cost with respect to the downloading cost. The download cost mainly reflects the operator's average CapEx or OpEx costs. The storage cost can be interpreted as memory usage cost or rental cost due to occupying the cache. Denote by α the weight of the storage cost with respect to the downloading cost, which is used to balance the storage-downloading costs. In general, the storage cost of unit data is no larger than the downloading cost from the server [5], [6]. Thus, $\alpha \in [0, 1]$. The operator adjusts the value of α according to the content demand for the server. For example, if the server is in high content demand, the operator expects helpers to offload. Thus, α may be set a low value, and vice versa. This is because a high content demand may lead to the redundant transmission of backhaul networks, thus resulting in the congestion of backhaul networks. In this case, reducing the storage cost by a low α enables helpers to store more contents, thus decreasing the burden of backhaul networks. Thus, the average of the storage and downloading costs over T time slots, denoted by $\mathrm{Cost}(\boldsymbol{x})$, is expressed as

$$\mathrm{Cost}(\boldsymbol{x}) = \frac{1}{T}\sum_{j\in\mathcal{J}} l_j [\underbrace{\sum_{k\in\mathcal{H}} \alpha f(x_{kj})}_{\text{storage cost}} + \underbrace{\sum_{i\in\mathcal{R}}\sum_{t=1}^{T} w_{ij} P_{ijt}}_{\text{downloading cost}}]. \tag{3}$$

$$D_{ijt} = \frac{v}{\mu\sqrt{\pi n_{ijt}}}\left[-n_{ijt}\mu^2(\mathrm{e}^{-\frac{(\frac{l_j}{v}-\frac{n_{ijt}}{\mu})^2}{4n_{ijt}\mu^2}} - \mathrm{e}^{-\frac{n_{ijt}}{4\mu^4}}) + n_{ijt}\sqrt{\pi n_{ijt}}\left(\phi(\frac{\mu l_j - v n_{ijt}}{v\mu^2\sqrt{2n_{ijt}}}) - \phi(\frac{-\sqrt{n_{ijt}}}{\sqrt{2}\mu^2})\right)\right] \\ + l_j\left(1 - \phi(\frac{l_j}{v\mu\sqrt{2n_{ijt}}} - \frac{\sqrt{2n_{ijt}}}{2\mu^2})\right) \tag{2}$$

C. Problem Formulation

Our Retention-aware caching problem (RACP) is to minimize $\text{Cost}(\boldsymbol{x})$ by optimizing $\boldsymbol{x}$. Thus, RACP is expressed in (4), where Eq. (4b) ensures that the total number of cached contents cannot exceed the cache capacity of helper k.

$$\min_{\boldsymbol{x}} \quad \text{Cost}(\boldsymbol{x}) \tag{4a}$$

$$\text{s.t.} \sum_{j\in\mathcal{J}} l_j 1_{\{x_{kj}>0\}} \le s_k, \quad k \in \mathcal{H} \tag{4b}$$

$$x_{kj} \in \{0,1,2,\dots,T\}, \quad k\in\mathcal{H},\ j\in\mathcal{J} \tag{4c}$$

D. Complexity Analysis

Theorem 1: ***RACP*** *is $\mathcal{NP}$-hard.*

Proof: See the Appendix. ■

III. Dynamic Programming Algorithm

As RACP is NP-hard, generally it is hard to derive its global optimum. However, in a symmetric case where $s_k = s$ for any helper k and $\lambda_{ik} = \lambda_i$ for any requester i, we can derive its global optimum. The assumption of uniform contact rate is in fact reasonable due to the following two reasons. First, there are some works that investigated this case and obtained some conclusions [33], [34]. For example, the work in [33] developed a class of spray routing protocols realizing excellent tradeoff between delays and low transmissions. Moreover, by analyzing the movement traces of 2000 taxies in *Shanghai* city, the contact rates of any two taxies are nearly uniform [35]. In the symmetric case, for any requester, the helpers holding a content are identical in performance, and hence it is unnecessary to care which helper stores which content. Rather, the caching performance depends only on the number of helpers that stores each content in each slot. Following the above intuition, denote by y_{jt} the number of helpers that stores content j in slot t, $y_{jt} \in \{0,1,\dots,H\}$. Here, $y_{jt} = 0$ denotes the case that there is no helper caching content j in slot t. In the remainder of this section, we focus on the symmetric case. The caching solution is denoted by $\boldsymbol{y}$, $\boldsymbol{y} = \{y_{jt}, j\in\mathcal{J}, t=1,2,\dots,T\}$, which is a $J\times T$ matrix of the J contents for the T slots. The expected amount of data that requester i collects from the encountered helpers after requesting content j in slot t, denoted by D'_{ijt}, is expressed in (5), as shown at the bottom of this page, where $n'_{ijt} = y_{jt}\lambda_i\theta$.

The average of the storage and downloading costs over T time slots, denoted by $\text{Cost}(\boldsymbol{y})$, is expressed as

$$\text{Cost}(\boldsymbol{y}) = \frac{1}{T}\sum_{j\in\mathcal{J}} l_j \left[\underbrace{\alpha\text{Cost}_\text{s}(\boldsymbol{y})}_{\text{storage cost}} + \underbrace{\text{Cost}_\text{d}(\boldsymbol{y})}_{\text{downloading cost}}\right], \tag{6}$$

where

$$\begin{cases} \text{Cost}_\text{s}(\boldsymbol{y}) = \sum\limits_{t=1}^{T}[f(t)-f(t-1)]y_{jt}, \\ \text{Cost}_\text{d}(\boldsymbol{y}) = \sum\limits_{i\in\mathcal{R}}\sum\limits_{t=1}^{T} w_{ij}P'_{ijt}. \end{cases} \tag{7}$$

In $\text{Cost}_\text{d}(\boldsymbol{y})$, P'_{ijt} denotes the network load probability which is the proportion of data that requester i has to download from the server in order to recover content j at the end of slot t in the symmetric case, with $P'_{ijt} = \frac{l_j - D'_{ijt}}{l_j}$. In $\text{Cost}_\text{s}(\boldsymbol{y})$, the expression of $(f(t)-f(t-1))y_{jt}$ expresses the storage cost using y_{jt} helpers to store content j in the t-th time slot. Thus, for any content j, $j\in\mathcal{J}$, after the summation over t, $t=1,2,\dots,T$, the result is the storage cost of the content.

Based on the above presentation, RACP is rewritten in (8), where Eq. (8b) guarantees that the total number of caching contents cannot exceed the total cache capacity, i.e., sH.

$$\min_{\boldsymbol{x'}} \quad \text{Cost}(\boldsymbol{y}) \tag{8a}$$

$$\text{s.t.}\ j\in\mathcal{J}\sum l_j y_{j1} \le sH, \tag{8b}$$

$$y_{jt}\in\{0,1,2,\dots,H\}, \quad j\in\mathcal{J},\ t=1,2,\dots,T \tag{8c}$$

A. Problem Analysis

The two storage costs of using k and k' helpers both grow over time, whereas the downloading cost does not change over time. If $k' > k$, then in any later time slot $t' > t$, the gap between the storage cost of using k' helpers and that of using k helpers becomes larger, whereas they produce the same downloading costs as in t. It is intuitive that if k helpers perform better than k' helpers in time slot t in terms of the total cost, it remains superior to k' helpers for any time slot $t' > t$. Therefore, we derive Lemma 2.

Lemma 2: For any content j, $j\in\mathcal{J}$, if k helpers minimize the total cost in time slot t, then for any later time slot, denoted by t' and t' > t, the total cost of using $k' > k$ helpers is greater than that of using k helpers.

$$D'_{ijt} = \frac{v}{\mu\sqrt{\pi n'_{ijt}}}\left[-n'_{ijt}\mu^2\left(\mathrm{e}^{-\frac{(\frac{l_j}{v}-\frac{n'_{ijt}}{\mu})^2}{4n'_{ijt}\mu^2}} - \mathrm{e}^{-\frac{n'_{ijt}}{4\mu^4}}\right) + n'_{ijt}\sqrt{\pi n'_{ijt}}\left(\phi\left(\frac{\mu l_j - v n'_{ijt}}{v\mu^2\sqrt{2n'_{ijt}}}\right) - \phi\left(\frac{-\sqrt{n'_{ijt}}}{\sqrt{2}\mu^2}\right)\right)\right]$$
$$+ l_j\left(1-\phi\left(\frac{l_j}{v\mu\sqrt{2n'_{ijt}}} - \frac{\sqrt{2n'_{ijt}}}{2\mu^2}\right)\right) \tag{5}$$

Proof: Denote by $\Delta(y_{jt})$ the cost for any content j and slot t, which is expressed as

$$\Delta(y_{jt}) = \sum_{i \in \mathcal{R}} w_{ij}(l_j - D'_{ijt}) + \alpha[f(t) - f(t-1)] l_j y_{jt}. \quad (9)$$

As the minimum of $\Delta(y_{jt})$ happens in $y_{jt} = k$, we have $\Delta(y'_{jt}) - \Delta(k) > 0$ for $y'_{jt} > k$. That is,

$$\sum_{i \in \mathcal{R}} w_{ij}(D^1_{ijt} - D^2_{ijt}) < \alpha[f(t) - f(t-1)] l_j (y'_{jt} - k), \quad (10)$$

where D^1_{ijt} and D^2_{ijt} denote the results of D'_{ijt} at k and y'_{jt}, respectively. In addition, as $f(t)$ is an increasing convex function in t, for $t' > t$, we have

$$\begin{aligned}\alpha[f(t) - f(t-1)] l_j (y'_{jt'} - k)& \\ < \alpha[f(t') - f(t'-1)] l_j (y'_{jt'} - k).& \quad (11)\end{aligned}$$

Combining (10) with (11), for $y'_{jt'} > k$, we have

$$\sum_{i \in \mathcal{R}} w_{ij}(D^1_{ijt} - D^2_{ijt}) < \alpha[f(t') - f(t'-1)] l_j (y'_{jt'} - k). \quad (12)$$

Note that $y'_{jt'} > k$ in time slot t' produces the same downloading cost as $y'_{jt} > k$ in t. By (12), we obtain $\Delta(y'_{jt'}) > \Delta(y_{jt'})$ for $y_{jt'} = k$ and $y'_{jt'} > k$. Hence the proof. ■

Denote by y^*_{jt} the optimal number of helpers for content j in time slot t. Next, we prove that for any content j, y^*_{jt} for all $t \geq 2$ can be obtained, given y^*_{j1}. Algorithm 1 describes the procedure.

Algorithm 1 Optimization for Given Initial Conditions

```
Require: $J$, $H$, and $T$
Ensure: $\mathbf{Y}$
1: for $j \leftarrow 1$ to $J$ do
2:   $\mathbf{Y}^j \leftarrow [0]_{(H+1)\times T}$
3:   if $l_j \leq s$ then
4:     for $y^*_{j1} \leftarrow 0$ to $H$ do
5:       $\mathbf{Y}^j_{y^*_{j1}+1,1} \leftarrow y^*_{j1}$
6:       for $t \leftarrow 2$ to $T$ do
7:         $y^*_{jt} \leftarrow \arg\min_{y_{jt} \in \{0,1,\dots,y^*_{j,t-1}\}} \{\Delta(y_{jt})\}$
8:         $\mathbf{Y}^j_{y^*_{j1}+1,t} \leftarrow \{y^*_{jt}\}$
9: return $\mathbf{Y}$
```

In Algorithm 1, denote by $\mathbf{Y}$ a three-dimensional matrix that is used to store the optimal solutions for all possible initial numbers of helpers. Denote by $\mathbf{Y}^j$ a $(H+1) \times T$ matrix that includes the optimal solutions of content j for all possible initial numbers of helpers. The entry $\mathbf{Y}^j_{y^*_{j1}+1,t}$ of $\mathbf{Y}^j$ denotes the optimum of content j for $t \geq 2$, given y^*_{j1}. Initially, all the entries of $\mathbf{Y}^j$ are set to be zero by Line 2, i.e., $\mathbf{Y}^j \leftarrow [0]_{(H+1)\times T}$. Line 4 takes into account all the possible y^*_{j1} in the range $[0, H]$. In Line 5, $\mathbf{Y}^j_{y^*_{j1}+1,1}$ is updated for a given y^*_{j1}. Lines 6-8 compute y^*_{jt} for $t \geq 2$, and then use y^*_{jt} to update $\mathbf{Y}^j_{y^*_{j1}+1,t}$. In Lines 6 and 7, the complexity of computing y^*_{jt} for all $t \geq 2$ is of $O(THR)$. The overall complexity of Algorithm 1 is of $O(JTH^2R)$.

Note that in Line 7, the algorithm derives y^*_{jt} for each time slot $t \geq 2$ with a greedy choice. Intuitively, the choice is globally optimal for given y^*_{j1}, however intuition is not rigor. Lemma 3 provides the formal optimality analysis.

Lemma 3: For any content j, given y^*_{j1}, Algorithm 1 gives the optimal number of helpers for all $t \geq 2$.

Proof: For any content, Algorithm 1 returns the numbers of helpers over all time slots, denoted by a sequence $y_1, y_2, \dots, y_T$. Suppose there is another monotonically decreasing sequence that is different from the first one, denoted by $y'_1, y'_2, \dots, y'_T$. Assume that the second sequence is superior to the first one given by the algorithm. Let y_t be the first element with $y_t < y'_t$, and suppose y_t remains less than the values of sequence two in consecutive time slots until time slot $t + n$ (for some $n \geq 0$). Namely, the elements $y'_t, y'_{t+1}, \dots y'_{t+n}$ of the second sequence are greater than y_t, whereas $y'_{t+n+1} \leq y_t$ for time slot $t + n + 1$. Suppose now we update the second sequence as follows. All of $y'_t, y'_{t+1}, \dots y'_{t+n}$ are updated to y_t, while the values of all other time slots in this sequence are kept. This is feasible because $y'_{t+n+1} \leq y_t$. Therefore, the updated sequence remains monotonically decreasing over t. Based on Lemma 2, the cost of the second sequence decreases due to the update, contradicting that the second sequence is superior to the first one. A special case is $t + n = T$, for which time slot $t+n+1$ does not exist. However the same update and conclusion hold.

One case remains. That is, there is no time slot t such that $y_t < y'_t$, yet the second sequence differs from the first one. In other words, $y_1 \geq y'_1$, $y_2 \geq y'_2$, $\dots$, $y_T \geq y'_T$. Suppose t is the first time slot with strict inequality, i.e., $y_t > y'_t$. Such t must exist, otherwise the two sequences will be identical to each other. Consider increasing the value of time t from y'_t to y_t, the second sequence remains feasible in terms of being monotonically decreasing. The reason is that t is the first time slot such that $y_t > y'_t$. Thus, $y_{t-1} = y'_{t-1}$ and $y_t \leq y'_{t-1}$. After setting y'_t to y_t, $y'_t \leq y'_{t-1}$ holds. The update reduces the cost of t, because when time slot t is considered by the algorithm, y_t leads to the minimum value. Therefore, in this case, the second sequence is not superior to the first one either. Hence the lemma. ■

By Algorithm 1, for any content j, y^*_{jt} can be derived for $t \geq 2$, given y^*_{j1}. Consequently, solving (8) is converted into finding the optimal number of helpers storing each content in slot one, i.e., y^*_{j1}, $j \in \mathcal{J}$. Therefore, the problem in (8) can be reformulated as in (13).

$$\min_{\boldsymbol{y_1}} \quad \frac{1}{T} \sum_{j \in \mathcal{J}} \mathrm{Cost}(\boldsymbol{y_j}) \quad (13a)$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{J}} l_j y_{j1} \leq sH, \quad (13b)$$

$$y_{j1} \in \{0, 1, 2, \dots, H\}, \quad j \in \mathcal{J} \quad (13c)$$

In (13a), $\boldsymbol{y_1} = [y_{11}, y_{21}, \dots, y_{J1}]$, and

$$\mathrm{Cost}(\boldsymbol{y_j}) = \alpha \sum_{t=1}^{T} [f(t) - f(t-1)] l_j y_{jt} + \sum_{i \in \mathcal{R}} \sum_{t=1}^{T} w_{ij}(l_j - D'_{ijt}),$$

where $\boldsymbol{y_j} = [y_{j1}, y^*_{j2}, \dots, y^*_{jT}]$. Note that for $\boldsymbol{y_j}$, the values of y^*_{jt} for $t \geq 2$ are induced by the first element of the vector. That is, they are optimal with respect to y_{j1}.

B. DP Algorithm

A dynamic programming (DP) algorithm is derived to obtain the optimum of y_{j1}, $j \in \mathcal{J}$. Denote by $a^*(c, j)$ the cost of the optimum with regard to the first j contents using c caching capacity units. Here, $a^*(0, j)$ denotes the cost that all the requested contents will be downloaded from the server as the available caching unit is zero. The value of $a^*(c, j)$ can be obtained via the recursive function in Lemma 4.

Lemma 4: The recursive function in (14) derives the value of $a^*(c, j)$.

$$a^*(c,j) = \begin{cases} \min\limits_{y_{j1} \in \{0,1,\dots,\min\{c,H\}\}} \{\mathit{Cost}(\boldsymbol{y_j}) + a^*(c - y_{j1}, j-1)\}, \\ \qquad 2 \leq j \leq J, \\ \min\limits_{y_{j1} \in \{0,1,\dots,\min\{c,H\}\}} \{\mathit{Cost}(\boldsymbol{y_j})\}, \\ \qquad j = 1. \end{cases} \tag{14}$$

Proof: Mathematical induction is used to prove the above lemma. First, when $j = 1$, for any c, the result is obvious. Then, suppose that $a^*(c, j)$ is the cost of the optimal solution for some j with respect to any c. By (14),

$$a^*(c, j+1) = \min_{x_{j+1,1} \in \{0,1,\dots,\min\{c,H\}\}} \{\mathrm{Cost}(\boldsymbol{y_{j+1}}) + a^*(c - y_{j+1,1}, j)\}.$$

For any $y_{j+1,1}$, $a^*(c - y_{j+1,1}, j)$ is the cost of the optimum and Cost$(\boldsymbol{y_{j+1}})$ is the cost of the optimum for content $j + 1$ (Lemma 3), thus together resulting in that $a^*(c, j + 1)$ is the cost of the optimal solution. Hence the conclusion. ■

C. Algorithm Summary and Complexity Analysis

The DP algorithm is described in Algorithm 2. The input parameters consist of $\mathbf{Y}$, J, s, and H. Here, $\mathbf{Y}$ is from the output of Algorithm 1. In Algorithm 2, $H' = sH$, and $\boldsymbol{g}_{cj}$ represents a vector that includes the optimal caching solutions of the first j contents under consideration using c caching capacity units. Initially, all the entries of matrix $\boldsymbol{y}$ are set to be zero by Line 1, i.e., $\boldsymbol{y} \leftarrow [0]_{J \times T}$. Lines 5 and 6 compute $a^*(c, 1)$ and y^*_{11}, respectively. In Line 7, y^*_{11} is stored in $\boldsymbol{g}_{c1}$. Lines 9 and 10 compute $a^*(c, j)$ and y^*_{j1} for $j \geq 2$, respectively. In Line 11, y^*_{j1} for $j \geq 2$ is stored in $\boldsymbol{g}_{cj}$. In Line 13, the j-th row of $\boldsymbol{y}$, denoted by $\boldsymbol{y}_j$, is updated using $\mathbf{Y}^j_{\boldsymbol{g}_{H'J}(j)+1}$, where $\boldsymbol{g}_{H'J}(j)$ denotes the j-th entry of vector $\boldsymbol{g}_{H'J}$.

Theorem 5: Algorithm 2 gives the global optimal solution of the problem in (8) in polynomial time.

Proof: Lemma 3 and Lemma 4 conclude the optimality. The computational complexity of Algorithm 2 is of $O(JsH^2)$. Note that, here s is not a parameter in terms of input size. However, the value of s is bounded by $\sum_{j \in \mathcal{J}} l_j$ due to the cache capacity constraint, otherwise the problem in (8) is easy to be solved without using the DP algorithm. Denote by $l_{\max}$ the value of $\max\limits_{j \in \mathcal{J}}\{l_j\}$. We have $\sum_{j \in \mathcal{J}} l_j \leq J l_{\max}$. In addition, Algorithm 2 depends on the output of Algorithm 1, i.e., $\mathbf{X}$. Thus, the overall complexity of obtaining the global optimal solution of the problem in (8) is of $O(\max\{J^2H^2, JTRH^2\})$, which is polynomial time complexity. Hence the proof. ■

Algorithm 2 The DP Algorithm

```
Require: $\mathbf{Y}$, $J$, and $H'$
Ensure: $\boldsymbol{y}$
1: $\boldsymbol{y} \leftarrow [0]_{J \times T}$
2: for $j = 1 : J$ do
3:   for $c = 0 : H'$ do
4:     if $j = 1$ then
5:       $a(c,1) \leftarrow \min\limits_{y_{11} \in \{0,1,\dots,\min\{\lfloor \frac{c}{l_j} \rfloor, H\}\}} \{\mathrm{Cost}(\boldsymbol{y_1})\}$
6:       $y^*_{11} \leftarrow \arg\min\limits_{y_{11} \in \{0,1,\dots,\min\{\lfloor \frac{c}{l_j} \rfloor, H\}\}} \{\mathrm{Cost}(\boldsymbol{y_1})\}$
7:       $\boldsymbol{g}_{c1} \leftarrow y^*_{11}$
8:     else
9:       $a(c,j) \leftarrow \min\limits_{y_{j1} \in \{0,1,\dots,\min\{\lfloor \frac{c}{l_j} \rfloor, H\}\}} \{\mathrm{Cost}(\boldsymbol{y_j}) + a(h - y_{j1}, j-1)\}$
10:      $y^*_{j1} \leftarrow \arg\min\limits_{y_{j1} \in \{0,1,\dots,\min\{\lfloor \frac{c}{l_j} \rfloor, H\}\}} \{\mathrm{Cost}(\boldsymbol{y_j}) + a(c - y_{j1}, j-1)\}$
11:      $\boldsymbol{g}_{cj} \leftarrow \boldsymbol{g}_{c - y^*_{j1}, j-1} \cup y^*_{j1}$
12: for $j = 1 : J$ do
13:   $\boldsymbol{y}_j \leftarrow \mathbf{Y}^j_{(\boldsymbol{g}_{H'J}(j)+1)}$
14: return $\boldsymbol{y}$
```

We remark that although the formulation of problem (8) exhibits a non-linear integer programming model, this class of problems is in fact tractable.

IV. Retention Aware Multi-Helper Caching Algorithm

For general cases, we propose a retention aware multi-helper caching algorithm (RAMA) to obtain a solution of RACP. In RAMA, the helpers are processed one by one. Initially, the retention time matrix, i.e., $\boldsymbol{x}$, is set as empty. In the first iteration, the retention time of each content for the first helper is optimized. Then, the optimized result for the first helper is fixed in later iterations when the optimization for the other helpers is performed. The optimization process is completed after processing last helper. The rationale of RAMA is that, in each iteration, the optimum of one row of the retention time matrix can be computed, with respect to the current values of the rest of the rows.

A. Main Observation

Theorem 6: The optimal caching solution of a helper, giving the caching solutions of the other $H - 1$ helpers, can be obtained in polynomial time.

Proof: Denote by $\mathbf{\Psi}$ a cost matrix, where its entry $\psi_{t+1,j}$ denotes the cost if the retention time of content j in the helper is t slots. The value of $\psi_{t+1,j}$ is computed using Eq. (3). Denote by $\mathbf{\Phi}$ an optimal cost matrix, where its entry $\phi_{u+1,q}$ denotes the cost of the optimum in respect of the first q contents using a cache size of u, $q \in \{1, 2, \dots, J\}$ and

$u \in \{0, 1, \ldots, s\}$, where s represents the cache capacity of the helper. The result of $\phi_{u+1,q}$ can be derived via a recursive function

$$\phi_{u+1,q} = \begin{cases} \min\limits_{\gamma}\{\psi'_{\gamma 1}\}, & \text{if } q = 1, \\ \min\limits_{\gamma}\{\psi'_{\gamma q} + \phi_{u+1-\gamma l_q, q-1}\}, & \text{else}, \end{cases} \tag{15}$$

where $\gamma \in \{0, 1\}$ and

$$\psi'_{\gamma q} = \begin{cases} \min\limits_{t=\{1,2,\ldots,T\}}\{\psi_{t+1,q}\}, & \gamma = 1, \\ \psi_{1q}, & \gamma = 0. \end{cases} \tag{16}$$

Giving the optimal cost of the $q-1$ contents, we can derive the optimal cost for up to q contents by (15) and (16), and this generalizes to cache size of s and all the J contents. Next, mathematical induction is used to prove the conclusion. When $q = 1$, $\phi_{u+1,1} = \min\limits_{\gamma}\{\psi'_{\gamma 1}\}$. If $\gamma = 1$, $\psi'_{11} = \min\limits_{t}\{\psi_{t+1,1}\}$ for $t = 1, \ldots, T$. Otherwise, $\psi'_{01} = \psi_{11}$. That is, there are $T + 1$ possible values. Comparing them one by one, the optimal value can be obtained. Suppose that $\phi_{u+1,r}$ is optimal for some r with any u. We will prove that $\phi_{u,r+1}$ is optimal. By (15), we have

$$\phi_{u+1,r+1} = \min_{\gamma}\{\psi'_{\gamma,r+1} + \phi_{u+1-\gamma,r}\}. \tag{17}$$

No matter $\gamma = 0$ or $\gamma = 1$, $\psi'_{\gamma,r+1}$ and $\phi_{u+1-\gamma,r}$ are optimal. Thus, $\phi_{u,r+1}$ is optimal.

The computational complexities for $\boldsymbol{\Psi}$ and $\boldsymbol{\Phi}$ are $\max\{O(RJ^2T^2), O(HTJ^2)\}$ and $O(JTs)$, respectively. Thus, the computational complexity for one helper is polynomial time, or $\max\{O(RJ^2T^2), O(HTJ^2)\}$, to be exact. ■

B. Algorithm Summary

Note that Theorem 6 applies even if some (or all) of the helpers' cache are empty. Thus allowing us to repeatedly use the theorem to successively construct a solution, with empty cache at the beginning. The resulting RAMA algorithm is described in Algorithm 3, in which the input parameters include H, J, T, $\boldsymbol{S}$, and $\boldsymbol{x}$. Here, $\boldsymbol{S} = \{s_1, s_2, \ldots, s_H\}$. Initially, all the entries of $\boldsymbol{x}$ are set to be zero, i.e., $\boldsymbol{x} = [0]_{H \times J}$. For a generic iteration for one helper, denote by r_q^* the optimal retention time of content q, and denote by $\boldsymbol{g}'_{uq}$ a vector consisting of the optimal retention time with the first q contents and cache size u. By Line 1, the helpers are treated one by one. By Line 2, $\boldsymbol{g}'$, $\boldsymbol{\Psi}$, and $\boldsymbol{\Phi}$ are initialized. Matrix $\boldsymbol{\Psi}$ is obtained by Lines 3-11. Lines 12-39 compute $\boldsymbol{\Phi}$ and $\boldsymbol{g}'_{uq}$. Line 40 use $\boldsymbol{g}'_{s_k J}$ to update the k-th row of $\boldsymbol{x}$, denoted by $\boldsymbol{x}^k$.

V. Performance Evaluations

We evaluate the performance of the proposed algorithms by considering two request distributions.

- Zipf distribution [9]–[14]: $w_{ij} = j^{-\gamma_i} / \sum_{k \in \mathcal{J}} k^{-\gamma_i}$, where γ_i is a Zipf parameter of requester i. Note that a larger γ_i means that the requester's interests are more focused on popular contents.

Algorithm 3 The RAMA Algorithm

```
Require: $h$, $J$, $T$, $\boldsymbol{S}$, and $\boldsymbol{x}$
Ensure: $\boldsymbol{x}$
 1: for $k \leftarrow 1$ to $H$ do
 2:   $\boldsymbol{g}' \leftarrow \emptyset$, $\boldsymbol{\Psi} \leftarrow [0]_{(T+1)\times J}$, and $\boldsymbol{\Phi} \leftarrow [0]_{(s_k+1)\times J}$
 3:   for $j \leftarrow 1$ to $J$ do
 4:     if $s_k \geq l_j$ then
 5:       for $t \leftarrow 0$ to $T$ do
 6:         $x_{kj} \leftarrow t$
 7:         $\psi_{t+1,j} \leftarrow \text{Cost}(\boldsymbol{x})$
 8:         $x_{kj} \leftarrow 0$
 9:     else
10:       $x_{kj} \leftarrow 0$
11:       $\psi_{1j} \leftarrow \text{Cost}(\boldsymbol{x})$
12:   for $q \leftarrow 1$ to $J$ do
13:     for $u \leftarrow 0$ to $s_k$ do
14:       if $q = 1$ then
15:         if $u = 0$ then
16:           $\phi_{u+1,1} \leftarrow \psi_{11}$, $\boldsymbol{g}'_{u1} \leftarrow 0$
17:         else
18:           if $u \geq l_q$ then
19:             $\phi_{u+1,1} \leftarrow \min\limits_{t\in\{0,1,\ldots,T\}}\{\psi_{t+1,1}\}$
20:             $\boldsymbol{g}'_{u1} \leftarrow \arg\min\limits_{t\in\{0,1,\ldots,T\}}\{\psi_{t+1,1}\}$
21:           else
22:             $\phi_{u+1,1} \leftarrow \psi_{11}$
23:             $\boldsymbol{g}'_{u1} \leftarrow 0$
24:       else
25:         if $u = 0$ then
26:           $\phi_{u+1,q} \leftarrow \psi_{1q} + \psi_{1,q-1}$,
27:           $\gamma^* \leftarrow 0$
28:         else
29:           if $u \geq l_q$ then
30:             $\phi_{u+1,q} \leftarrow \min\limits_{\gamma\in\{0,1\}}\{\psi'_{\gamma q} + \phi_{u+1-\gamma l_q,q-1}\}$,
             $\psi'_{\gamma q} \leftarrow \min\limits_{t\in\{1,2,\ldots,T\}}\{\psi_{t+1,q}\}$ if $\gamma = 1$, otherwise
             $\psi'_{0q} \leftarrow \psi_{1q}$
31:             $\gamma^* \leftarrow \arg\min\limits_{\gamma\in\{0,1\}\}}\{\psi'_{\gamma q} + \phi_{u+1-\gamma l_q,q-1}$
32:           else
33:             $\phi_{u+1,q} \leftarrow \psi_{1q} + \psi_{u+1,q-1}$,
34:             $\gamma^* \leftarrow 0$
35:         if $\gamma^* = 0$ then
36:           $r_q^* \leftarrow 0$
37:         else
38:           $r_q^* \leftarrow \arg\min\limits_{t\in\{1,2,\ldots,T\}}\{\psi_{t+1,q}\}$
39:         $\boldsymbol{g}'_{uq} \leftarrow \boldsymbol{g}'_{u-\gamma^*,q-1} \cup \{r_q^*\}$
40:   $\boldsymbol{x}^k \leftarrow \boldsymbol{g}'_{s_k J}$
41: return $\boldsymbol{x}$
```

- Exponential distribution [27]: $w_{ij} = \frac{e^{-j}}{\sum_{k\in\mathcal{J}} e^{-k}}$ for each requester i. In this distribution, the requesters' interests are concentrated on popular contents. The gaps between the high popular contents and low popular contents of this distribution are larger than that of the Zipf distribution.

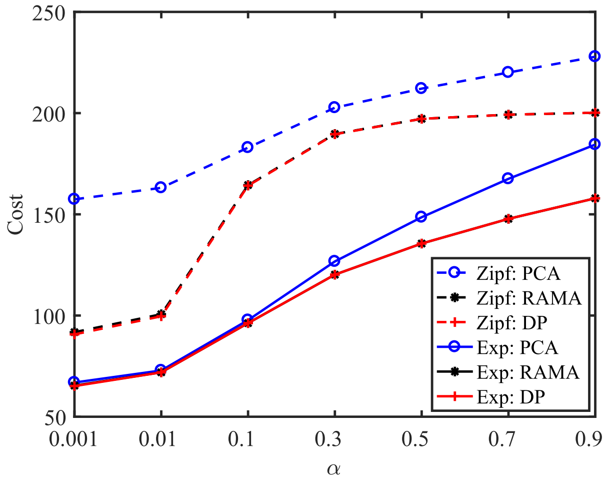

Fig. 3. Impact of α in the symmetric case where $R = 20$, $H = 5$, $J = 20$, $s = 40$ MB, and $T = 5$.

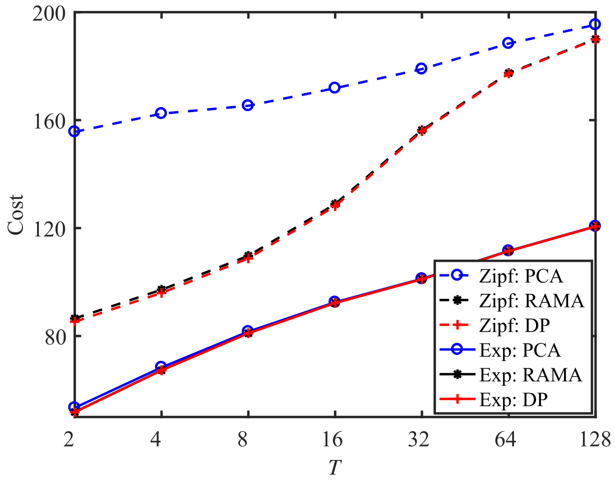

Fig. 4. Impact of T in the symmetric case where $R = 20$, $H = 5$, $J = 20$, $s = 40$ MB, and $\alpha = 0.01$.

In the numerical results, in addition to DP and RAMA, we include the following two algorithms.

- Random caching algorithm (RCA): The algorithm considers helpers one by one. Initially, all the helpers' caches are empty. For the first helper, the contents are stored randomly with the same probability, until the cache capacity units of the helper are used up. The algorithm optimizes the retention time of each cached content in the first helper by comparing T possible values (i.e., from 1 to T), and then chooses the best. The retention time of each cached content in this helper is fixed when the algorithm performs the optimization for other helpers. The algorithm ends until this process is complete for the last helper.
- Popularity-based caching algorithm (PCA): The algorithm considers helpers one by one. Each helper stores contents according to the content's request probability until its cache capacity units are used up. The optimization process of the retention time of each cached content is the same as that in random caching. In the algorithm, all the helpers store the same contents if their cache sizes are the same, while the retention times of cached contents may differ.

In [36], the experiments show that the contact rate can be characterized by a Gamma distribution. Based on this fact, the contact rate between requester i and helper k, λ_{ik}, is generated by a Gamma distribution $\Gamma(44.3, 1/1088)$. The mean data transmission rate, v, is set to 20 Mbps [27]. We set $T \times \theta = 24$ hours [4], i.e., one day.

A. Symmetric Case

Figs. 3 and 4 show the impacts of α and T on the total cost in the symmetric case. As presented in Section III, DP can obtain the global optimum in this case. From Figs. 3 and 4, we make the following observations. First, in Fig. 3, when $\alpha \leq 0.3$, the impact of α is significant for both DP and RAMA. When $\alpha > 0.3$, its impact gradually weakens for DP and RAMA, especially for the case of Zipf distribution. This is explained by the fact that with the increase of α, the storage price per unit grows, and the total number of cached contents decreases. When $\alpha \leq 0.3$, the reduction in storage cost due to fewer cached contents does not compensate the cost growth because of high storage price per unit, leading to that the storage cost increases. In addition, the downloading cost slowly increases due to the decrease of the number of cached contents. Thus, the total cost consisting of the storage cost and the downloading cost quickly grows. When $\alpha > 0.3$, due to the high storage price per unit, the helpers will choose caching a short time duration or even no caching, resulting in that the downloading cost becomes large and the storage cost becomes small. But, the increase in the former is larger than the decrease in the latter, and this phenomenon weakens for big α. Second, in Fig. 4, the total cost increases with respect to T. The explanation is that when T becomes large, the time duration of one slot, i.e., θ, shortens. That is, the delay tolerance duration of each content becomes small. As a result, the requesters have not enough time to collect data from the helpers, giving rise to high downloading cost, but the change of the storage cost is subtle. Therefore, the two cost types together cause the growth of the total cost. Finally, and most importantly, it can be observed that for RAMA, in Fig. 3 the solutions of RAMA and DP almost completely overlap, and the gap between them is less than 1.22%. In Fig. 4, for RAMA, the gap to the global optimum is less than 2%. This observation shows that the solution of RAMA is very close to the global optimum. Note that RCA is not shown in the figures, as the performance is extremely poor in comparison to RAMA.

B. General Case

In the more general problem setup, λ_{ik} is generated by a Gamma distribution, and the cache size differs by vehicles. The experiments in [9] manifested that the average contact rate is linearly proportional to the movement speed; a high speed results in a large contact rate. Based on this fact, investigating the impact of movement speed can be done by examining the effect of contact rate. The contact rate between requester i and

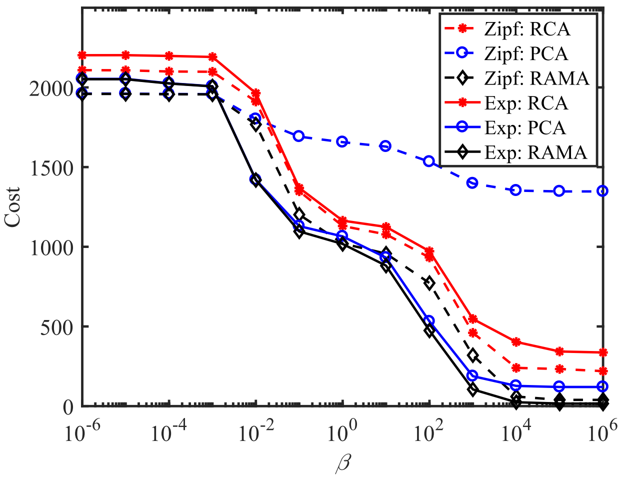

Fig. 5. Impact of β when $R = 80$, $H = 20$, $J = 20$, $\alpha = 0.01$, and $T = 8$.

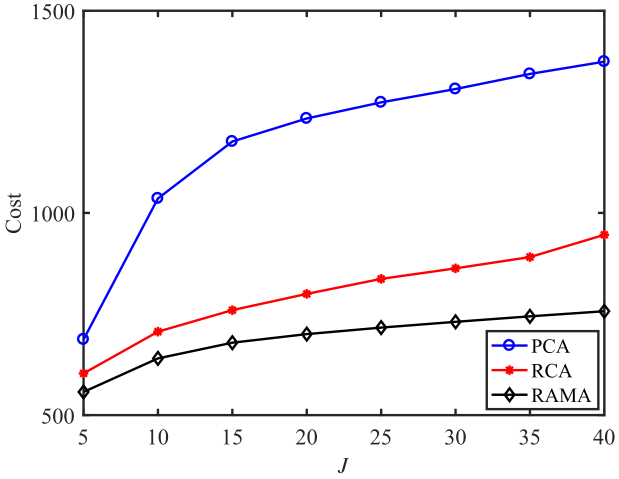

Fig. 6. Impact of J when $R = 80$, $H = 20$, $\alpha = 0.01$, and $T = 8$.

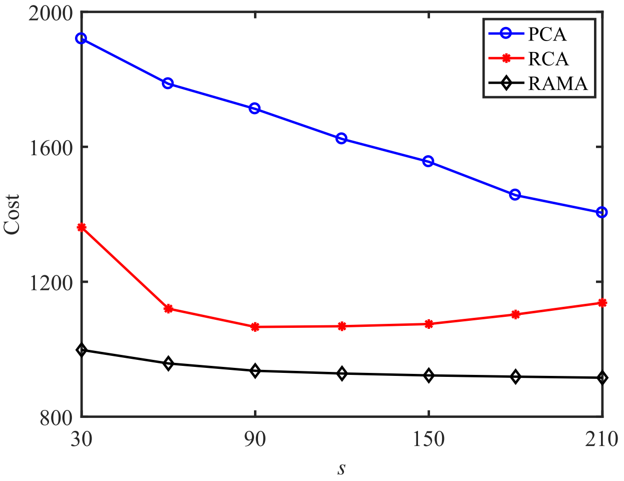

Fig. 7. Impact of s when $R = 80$, $H = 20$, $J = 20$, $\alpha = 0.01$, and $T = 8$.

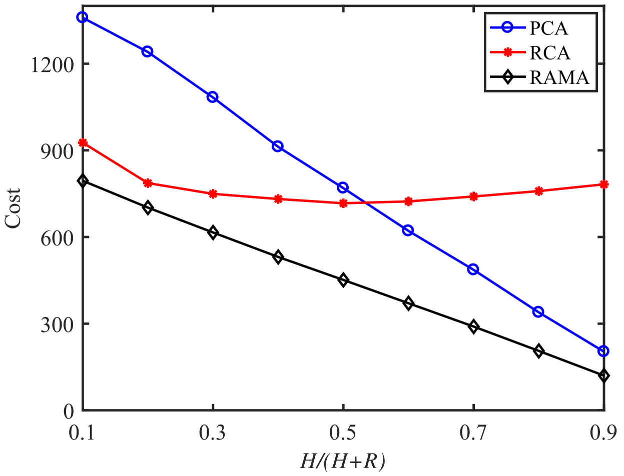

Fig. 8. Impact of $\frac{H}{H+R}$ when $R + H = 100$, $\alpha = 0.01$, $s = 80$, $J = 20$, and $T = 8$.

helper k, λ_{ik}, is generated by a Gamma distribution $\Gamma(\beta, \delta)$, where $\beta\delta$ is equal to the average contact rate. Fig. 5 explores the impact of β via fixing δ. First, it can be observed that when $10^{-3} < \beta < 10^{4}$, the total cost decreases as each requester is likely to collect more needed data from the helpers. When $\beta \leq 10^{-3}$ and $\beta \geq 10^{4}$, the total cost almost stays constant. The observation is explained by the fact that when $\beta \leq 10^{-3}$, the contact rates are very low. Hence, the helpers provide very little help. In such a case, the helpers should store the popular contents. As RCA stores a part of infrequently requested contents, its performance is inferior than that of RAMA and PCA. When $\beta \geq 10^{4}$, each requester has enough opportunity to access the cached contents of all the helpers, and the total cost stays constant due to the limitations of cache capacity and the number of helpers, etc. In addition, one discovers that when $\beta > 10^{-2}$, the total cost by PCA is apparently larger than that by both RCA and RAMA for the case of Zipf distribution as the helpers in both RCA and RAMA avoid excessive duplication caching.

Fig. 6 analyzes the impact of J on the total cost for the case of Zipf distribution. One observes that RAMA outperforms the other two algorithms. More importantly, when J grows, the improvement is very apparent. By increasing J from 5 to 40, the improvement over RCA grows from 7.6% to 19.98%, and the improvement over PCA grows from 18.89% to 44.92%. The reason is that for RCA, increasing J directly results in higher diversity of contents, thus growing the probability that more unpopular contents are stored in the helpers. For PCA, as the stored contents of all the helpers do not change with respect to J, increasing J leads to considerable growth. For RAMA, as the caching solution of each helper is optimized, the increase in the total cost with respect to J is less than that in RCA and PCA. Hence, the improvements over RCA and PCA become large. Note that J denotes the system size. Therefore, RAMA is beneficial to large-scale scenarios.

Fig. 7 and Fig. 8 report the impact of cache capacity on the total cost by changing cache size per helper and the proportion of the number of helpers, respectively. In the two figures, the requests follow a Zipf distribution. In Fig. 7, the sizes of all the caches are set to be the same, i.e., $s_k = s$ for any k. One observes that for RAMA and PCA, the total cost decreases with the growth of s or $\frac{H}{H+R}$ as more requested

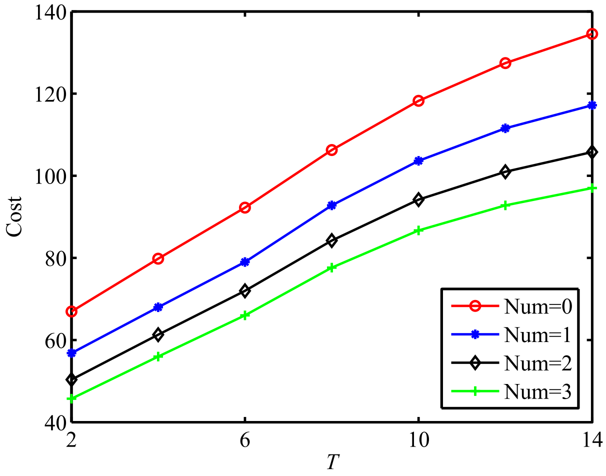

Fig. 9. Impact of caching at RSUs.

contents can be accommodated by caching. For RCA, the total cost first goes down and then grows again. This is because, when $s \leq 90$ or $\frac{H}{H+R} \leq 0.5$, the decrease in the downloading cost is larger than the increase in the storage cost, whereas for $s > 90$ or $\frac{H}{H+R} > 0.5$, the reason is the opposite due to caching more unpopular contents.

C. Impact of RSUs

In our system model, for tractability, we consider caching at vehicles. In this subsection, we briefly evaluate the impact of caching at both RSUs and helpers on caching performance by Fig. 9. In this figure, all the RSUs store the same contents that are the most popular, and the label "*Num*" denotes the number of contents that are stored at an RSU. If RSUs store a content, it is no longer necessary to store the content at any helper. This is because RSUs and helpers are all edge devices. The purpose of caching at edge devices is to not only reduce backhaul burden, but also reduce the content downloaded delay. If RSUs store a content, the requesters interested in the content will directly download the content from RSUs due to a shorter time compared to the waiting time in order to come into contact with helpers. Note that in our paper, the backhaul burden is represented as the network load probability (i.e., P_{ijt}). It can be observed that with the increase of Num, the total cost decreases. The reason is that the network load probability decreases with the increase of Num, thus reducing the downloading cost.

So far, our numerical results have not considered the cases that the RSUs' links may be congested. In addition, the results have not considered the downloading cost due to using RSUs or helpers. In this subsection, we use one setup to consider the impact of the two aspects on caching performance by Fig. 10. In Fig. 10, all the RSUs are assumed to store only the most popular content. Here, a symmetric case is investigated, because the proposed dynamic programming algorithm can derive the global optimal caching solutions. The ratios of the downloading cost of unit data due to the downloading operation from RSUs and helpers with respect to that from the server are assumed to be 0.66 and 0.033 [13], respectively.

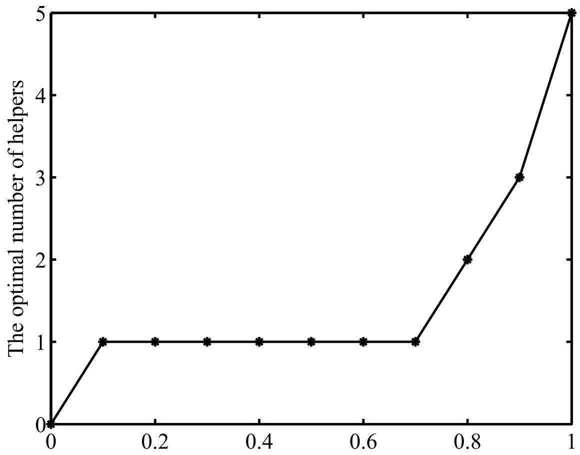

Fig. 10. Impact of congestion when $R = 5$, $H = 10$, $J = 20$, and $\alpha = 0.001$.

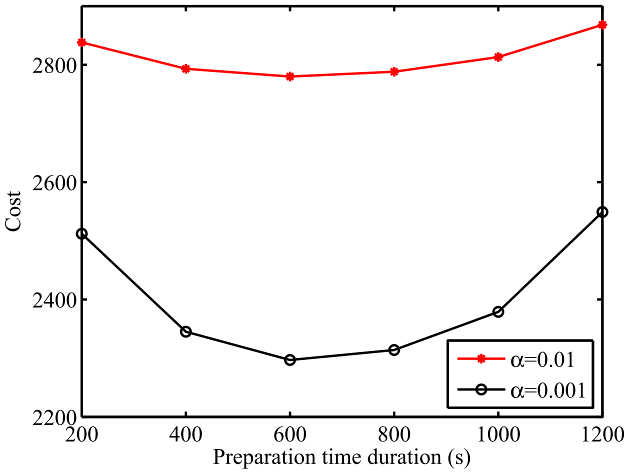

Fig. 11. Impact of the preparation time duration.

In this figure, the X-axis represents the proportion of data of the content that cannot be downloaded due to congestion, which is a direct consequence of the severeness of congestion. When X-axis is zero, there is no congestion. When X-axis is one, the RSUs are too congested to contribute to downloading the content. The Y-axis denotes the optimal number of helpers storing the content in the first time slot. It can be observed that when there is no congestion, the optimal number of helpers is zero. The reason is that the requesters can directly download the complete content from RSUs in a time slot. Thus, it is unnecessary to store the content at any helper. When X-axis is in the range of $[0.1, 0.7]$, the optimal performance is achieved by using $1/10$ of the helpers to store the content. When congestion becomes more and more severe, the optimal number of helpers increases. Therefore, by this figure, we can learn that indeed, RSUs' congestion directly affects the optimal number of helpers storing the content.

D. Impact of the Preparation Time Duration

To place contents on the helpers, there is a preparation time duration in which the helpers may download contents from the server. For the previous results, we have assumed that the preparation time duration is long enough, so that a helper can download the contents that it will store. However, the need of preparation, depending on its length, may impact performance such that only some of the intended contents are downloaded by a helper. Thus, in this subsection, we consider a more stringent preparation time duration, and investigate the impact of its length on caching performance, see Fig. 11. In Fig. 11, all the contents have the same size, with $l_j = 400$ MB for any content j, $j \in \mathcal{J}$. All the helpers have the same caching capacity, with $s_k = 2$ GB for any helper k, $k \in \mathcal{H}$. The average downloading rate from a base station to a helper is set to be 2 MB/s [37]. The time duration of the time period is 1800 s. The total time period is 1800 s. The number of time slots is 8, i.e., $T = 8$. In Fig. 11, it can be observed that with the increase of the preparation time duration, the total cost first decreases, and then increases. The reason is that although the helpers can download more contents with longer preparation time, the time duration of each time slot decreases, that is, less time is available to the requesters for collecting contents from the helpers. Thus, there is a tradeoff with respect to the preparation time duration.

VI. Conclusions

This paper has modeled a retention-aware caching problem in vehicular networks, i.e., RACP. In the model, the contact between any two vehicles follows a Poisson distribution and the communication time duration of one contact follows an exponential distribution. The hardness of RACP has been proved. For problem-solving, an optimal dynamic programming algorithm, DP, has been developed to derive the problem's optimum for symmetric cases. For general cases, a retention aware multi-helper caching algorithm, RAMA, has been proposed to satisfactorily solve the problem. Performance evaluations show that RAMA achieves near-optimal performance by comparing it to DP in the symmetric case. In addition, RAMA leads to significant improvement with respect to random caching and popularity-based caching.

An extension of our work is to consider the case of updating the cached contents in each slot. This can be modeled as to minimize the sum of the updating cost, the storage cost, and the downloading cost, with a constraint on cache capacity. Another extension of this paper is the consideration of caching at both vehicles and road side units.

Appendix

We use a polynomial-time reduction from the 3-SAT problem that is NP-complete [21]. Consider a 3-SAT problem with n clauses and m Boolean variables. The Boolean variables are denoted by $y_1, y_2, \ldots, y_m$. A literal represents a Boolean variable or its negation. The negation of y_i is denoted by $\hat{y}_i$, $i \in \{1, 2, \ldots, m\}$. The 3-SAT problem asks if the set of n clauses, each consisting of exact three different literals (e.g., $\hat{y}_1 \vee y_2 \vee y_3$), can be satisfiable simultaneously by a true/false assignment of the Boolean variables.

A reduction from the 3-SAT problem is constructed as follows. There are $3m + n$ vehicles, divided into three sets, namely, $2m$ literal vehicles (helpers), m auxiliary vehicles (requesters), and n clause vehicles (requesters). There are two contents, a and b, both of size one. Each literal helper's cache size is one. The cache size of all clause requester's cache size and auxiliary requester's cache size is zero. We set $T = 1$, $\theta = 1$, $\mu = 10$, and $v = 1$. The value of α is very small, thus guaranteeing that storing a content induces virtually no cost. The literal helpers constitute m pairs. We set $\lambda_{ik} = \ln(\frac{1}{\epsilon})$ for helpers i and k in each of the m pairs, where ϵ is a very small positive number. We also set $\lambda_{ik} = \ln(\frac{1}{1-\epsilon})$ for helpers i and k that are from different pairs. For a moment, suppose that one of the helpers in each pair stores content a, and the other helper stores content b, or vice versa. Namely, the cached content of any pair is either ab or ba, which is equivalent to the Boolean value assignment in the 3-SAT instance.

The probabilities that each auxiliary requester requests contents a and b are the same. For the i-th auxiliary requester and the k-th literal helper pair, we set $\lambda_{ik} = \ln(\frac{1}{\epsilon})$ if $i = k$. Otherwise, we set $\lambda_{ik} = \ln(\frac{1}{1-\epsilon})$. The total downloading cost for the auxiliary requesters is $m(1 - D(z_0))$, where $z_0 = \ln(\frac{1}{\epsilon(1-\epsilon)^{m-1}})$ and $D(z)$ is expressed in (18), as shown at the bottom of this page.

Each clause requester requests content a with probability one. For the two clause requesters i and k, λ_{ik} can be anything due to a cache capacity of zero. If i is a clause requester and k is one of the three literal helpers corresponding to the three literals of this clause, we set $\lambda_{ik} = \ln(\frac{1}{\epsilon})$. Otherwise, $\lambda_{ik} = \ln(\frac{1}{1-\epsilon})$. If content a is stored at least one of the three literal helpers, the downloading cost for a clause requester is at most $1 - D(z_1)$, where $z_1 = \ln\frac{1}{\epsilon(1-\epsilon)^{m-3}}$. For all the clause requesters, the corresponding total downloading cost is at most $n(1 - D(z_1))$. The storage cost is $2m\alpha$. Therefore, the total cost is at most $\Delta_s \triangleq m(1 - D(z_0)) + n(1 - D(z_1)) + 2m\alpha$.

By the above construction, the total cost is no more than Δ_s if the 3-SAT instance is satisfiable. Otherwise, at least one clause requester has in fact no other choice but to download the content from the server. In such a case, the total cost is at least $m(1-D(z_0))+(n-1)(1-D(z_2))+(1-D(z_3))+2m\alpha$, where $z_2 = \ln(\frac{1}{\epsilon^3(1-\epsilon)^{m-3}})$ and $z_3 = \ln(\frac{1}{(1-\epsilon)^{m-3}})$, which is larger than Δ_s. Therefore, whether or not there exists a caching decision with a total cost of no more than Δ_s gives the correct answer to 3-SAT.

$$D(z) = \frac{\phi(\frac{1}{10\sqrt{2z}} - \frac{100\sqrt{2z}}{2})}{20\sqrt{\pi z}}[2 - 200z(\mathrm{e}^{-\frac{z}{40000}} + \mathrm{e}^{-\frac{(10-z)^2}{40000z}}) + z\sqrt{\pi z}(\phi(\frac{10-z}{100\sqrt{2z}}) - \phi(-\frac{\sqrt{z}}{100\sqrt{2}}))] + 1 - \phi(\frac{1}{10\sqrt{2z}} - \frac{\sqrt{2z}}{200}) \quad (18)$$

Next, the case that remains to be considered is that the same content is stored in some of the literal helper pairs. Assume that exact one pair stores contents aa or bb, and the other pairs store ab or ba. In such a case, the downloading cost for all the auxiliary requesters is $(m-1)(1-D(z_0))+\frac{1}{2}(1-D(z_4))+\frac{1}{2}(1-D(z_5))$, where $z_4=\ln(\frac{1}{\epsilon^2(1-\epsilon)^{m-2}})$ and $z_5=\ln(\frac{1}{(1-\epsilon)^{m-2}})$. For all the clause requesters, if they can obtain the data of content a from the literal helpers, their downloading cost is no less than $(n-1)(1-D(z_2))+(1-D(z_6))$, where $z_6=\ln(\frac{1}{\epsilon^2(1-\epsilon)^{m-3}})$. Thus, the total cost is at least $(m-1)(1-D(z_0))+\frac{1}{2}(1-D(z_4))+\frac{1}{2}(1-D(z_5))+(n-1)(1-D(z_2))+(1-D(z_6))+2m\alpha$, which is larger than Δ_s. If more than one pair stores the same contents aa or bb, the total cost will be even higher. Therefore, the previous conclusion also holds.

From the above, solving RACP will solve the 3-SAT problem which is NP-complete. Hence the conclusion.

References

[1] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. M. Leung, "Cache in the air: Exploiting content caching and delivery techniques for 5G systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 131–139, Feb. 2014.

[2] P. Fan, "Coping with the big data: Convergence of communications, computing and storage," *China Commun.*, vol. 13, no. 9, pp. 203–207, Sep. 2016.

[3] X. Lai, J. Xia, M. Tang, H. Zhang, and J. Zhao, "Cache-aided multiuser cognitive relay networks with outdated channel state information," *IEEE Access*, vol. 6, pp. 21879–21887, 2018.

[4] S. Shukla and A. A. Abouzeid, "Proactive retention aware caching," in *Proc. IEEE Infocom*, May 2017, pp. 1–9.

[5] S. Shukla, O. Bhardwaj, A. A. Abouzeid, T. Salonidis, and T. He, "Hold'em caching: Proactive retention-aware caching with multi-path routing for wireless edge networks," in *Proc. ACM Mobihoc*, 2017, pp. 1–10.

[6] S. Shukla, O. Bhardwaj, A. A. Abouzeid, T. Salonidis, and T. He, "Proactive retention-aware caching with multi-path routing for wireless edge networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1286–1299, Jun. 2018.

[7] S. Shukla and A. A. Abouzeid, "Optimal device-aware caching," *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 1994–2007, Jul. 2017.

[8] R. Wang, X. Peng, J. Zhang, and K. B. Letaief, "Mobility-aware caching for content-centric wireless networks: Modeling and methodology," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 77–83, Aug. 2016.

[9] R. Wang, J. Zhang, S. H. Song, and K. B. Letaief, "Mobility-aware caching in D2D networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 8, pp. 5001–5015, Aug. 2017.

[10] R. Wang, J. Zhang, S. H. Song, and K. B. Letaief, "Mobility increases the data offloading ratio in D2D caching networks," in *Proc. ICC*, 2017, pp. 1–6.

[11] R. Wang, J. Zhang, and K. B. Letaief, "Incentive mechanism design for cache-assisted D2D communications: A mobility-aware approach," in *Proc. SPAWC*, 2017, pp. 1–6.

[12] T. Deng, G. Ahani, P. Fan, and D. Yuan, "Cost-optimal caching for D2D networks with presence of user mobility," in *Proc. IEEE GLOBECOM*, Dec. 2017, pp. 1–6.

[13] T. Deng, G. Ahani, P. Fan, and D. Yuan, "Cost-optimal caching for D2D networks with user mobility: Modeling, analysis, and computational approaches," *IEEE Trans. Wireless Commun.*, vol. 17, no. 5, pp. 3082–3094, May 2018.

[14] T. Deng, L. You, P. Fan, and D. Yuan, "Device caching for network offloading: Delay minimization with presence of user mobility," *IEEE Wireless Commun. Lett.*, vol. 7, no. 4, pp. 558–561, Aug. 2018.

[15] M. Chen, Y. Hao, L. Hu, K. Huang, and V. Lau, "Green and mobility-aware caching in 5G networks," *IEEE Trans. Wireless Commun.*, vol. 16, no. 12, pp. 8347–8361, Dec. 2017.

[16] G. Ahani and D. Yuan, "On optimal proactive and retention-aware caching with user mobility," in *Proc. IEEE VTC-Fall*, Aug. 2018, pp. 1–5.

[17] *Study on Scenarios and Requirements for Next Generation Access Technologies (Release 14)*, document 3GPP TR 38.913, 3GPP 3rd Generation Partnership Project; Technical Specification Group Radio Access Network, 2016.

[18] X. Cheng, C. Chen, W. Zhang, and Y. Yang, "5G-Enabled cooperative intelligent vehicular (5GenCIV) framework: When benz meets marconi," *IEEE Intell. Syst.*, vol. 32, no. 3, pp. 53–59, May 2017.

[19] S. Chen *et al.*, "Vehicle-to-everything (V2X) services supported by LTE-based systems and 5G," *IEEE Commun. Standards Mag.*, vol. 1, no. 2, pp. 70–76, Jun. 2017.

[20] S. Chen, J. Hu, Y. Shi, and L. Zhao, "LTE-V: A TD-LTE-based V2X solution for future vehicular network," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 997–1005, Dec. 2016.

[21] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1979.

[22] H. Luo, X. Meng, R. Ramjee, P. Sinha, and L. Li, "The design and evaluation of unified cellular and ad-hoc networks," *IEEE Trans. Mobile Comput.*, vol. 6, no. 9, pp. 1060–1074, Sep. 2007.

[23] X. Zhuo, W. Gao, G. Cao, and Y. Dai, "Win-coupon: An incentive framework for 3G traffic offloading," in *Proc. IEEE ICNP*, Oct. 2011, pp. 206–215.

[24] *Intelligent Transport System (ITS); Vehicular Communications; Basic Set of Applications; Definitions, V1.1.1.*, document ETSI TR 201 638, Mar. 2009.

[25] *Study on LTE-Based V2X Services, Version 1.0.0.*, document 36.885, 3GPP, Mar. 2016.

[26] *Family of Standards for Wireless Access in Vehicular Environments (WAVES)*, IEEE Standard 1609, U.S. Department of Transportation, Apr. 2013.

[27] X. Zhu, Y. Li, D. Jin, and J. Lu, "Contact-aware optimal resource allocation for mobile data offloading in opportunistic vehicular networks," *IEEE Trans. Veh. Tech.*, vol. 66, no. 8, pp. 7384–7399, Aug. 2017.

[28] C. Wang, Y. Li, and D. Jin, "Mobility-assisted opportunistic computation offloading," *IEEE Commun. Lett.*, vol. 18, no. 10, pp. 1779–1782, Oct. 2014.

[29] X. Cheng, L. Yang, and X. Shen, "D2D for intelligent transportation systems: A feasibility study," *IEEE Trans. Intell. Trans. Syst.*, vol. 16, no. 4, pp. 1784–1793, Jan. 2015.

[30] P. Sermpezis and T. Spyropoulos, "Modelling and analysis of communication traffic heterogeneity in opportunistic networks," *IEEE Trans. Mobile Comput.*, vol. 14, no. 11, pp. 2316–2331, Nov. 2015.

[31] H. Zhu, L. Fu, G. Xue, Y. Zhu, M. Li, and L. Ni, "Recognizing exponential inter-contact time in VANETs," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–5.

[32] Y. Li, D. Jin, L. Zeng, and S. Chen, "Revealing patterns of opportunistic contact durations and intervals for large scale urban vehicular mobility," in *Proc. IEEE ICC*, Jun. 2013, pp. 1646–1650.

[33] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Efficient routing in intermittently connected mobile networks: The multiple-copy case," *IEEE/ACM Trans. Netw.*, vol. 16, no. 1, pp. 77–90, Feb. 2008.

[34] X. Zhang, G. Neglia, J. Kurose, and D. Towsley, "Performance modeling of epidemic routing," *Comput. Netw.*, vol. 51, nos. 10–11, pp. 2867–2891, Jul. 2007.

[35] Y. Li, G. Su, P. Hui, D. Jin, L. Su, and L. Zeng, "Multiple mobile data offloading through delay tolerant networks," in *Proc. ACM CHANTS*, 2011, pp. 1–6.

[36] A. Passarella and M. Conti, "Analysis of individual pair and aggregate intercontact times in heterogeneous opportunistic networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 12, pp. 2483–2495, Dec. 2013.

[37] C. Cox, *An Introduction to LTE: LTE, LTE-Advanced, SAE and 4G Mobile Communications*, 1st ed. London. U.K.: Wiley, 2012.

Tao Deng (S'14) received the B.Eng. degree from Henan Normal University, Xinxiang, China, in 2012. He is currently pursuing the Ph.D. degree with the Key Laboratory of Information Coding and Transmission, School of Information Science and Technology, Southwest Jiaotong University (SWJTU), Chengdu, China.

From 2016 to 2018, he was a Visiting Ph.D. Student with the Department of Information Technology, Uppsala University, Sweden. His research interests include 5G wireless caching, mobility management, and performance modeling for wireless networks.

Pingzhi Fan (F'15) received the M.Sc. degree in computer science from Southwest Jiaotong University, China, in 1987, and the Ph.D. degree in electronic engineering from Hull University, U.K., in 1994. He was a Visiting Professor with Leeds University, U.K., in 1997, and a Guest Professor with Shanghai Jiaotong University, in 1999. He is currently a Distinguished Professor and the Director of the Institute of Mobile Communications, Southwest Jiaotong University. He has published over 290 international journal papers and eight books (incl. edited), and is the inventor of 27 granted patents. His current research interests include vehicular communications, massive multiple access, and coding techniques, etc. He is a Fellow of IET, CIE, and CIC. He was a recipient of the UK ORS Award in 1992, the NSFC Outstanding Young Scientist Award in 1998, the IEEE VTS Jack Neubauer Memorial Award in 2018, and the IEEE Signal Processing Society Paper Award in 2019. He served as the General Chair or the TPC Chair for a number of international conferences, including VTC'2016 Spring, IWSDA'2019, and ITW'2018. He is the Founding Chair of the IEEE Chengdu (CD) Section, the IEEE VTS BJ Chapter, and the IEEE ComSoc CD Chapter. He also served as an EXCOM Member for the IEEE Region 10, IET (IEE) Council, and IET Asia–Pacific Region. He was an IEEE VTS Distinguished Lecturer from 2015 to 2019.

Di Yuan (SM'16) received the M.Sc. degree in computer science and engineering and the Ph.D. degree in optimization from the Linkoping Institute of Technology, in 1996 and 2001, respectively. After his Ph.D., he has been an Associate Professor and then a Full Professor with the Department of Science and Technology, Linkoping University, Sweden. In 2016, he joined Uppsala University, Sweden, as a Chair Professor. He was a Guest Professor with the Technical University of Milan (Politecnico di Milano), Italy, in 2008, and a Senior Visiting Scientist with Ranplan Wireless Network Design Ltd., U.K., in 2009 and 2012. In 2011 and 2013, he has been working part-time with Ericsson Research, Sweden. In 2014 and 2015, he was a Visiting Professor with the University of Maryland, College Park, MD, USA. He has been in the Management Committee of four European Cooperation in field of Scientific and Technical Research (COST) actions, Invited Lecturer of European Network of Excellence EuroNF, and Principal Investigator of several European FP7 and Horizon 2020 projects. His current research interests include network optimization of 4G and 5G systems and capacity optimization of wireless networks. He was a co-recipient of the IEEE ICC 2012 Best Paper Award, and a Supervisor of the Best Student Journal Paper Award by the IEEE Sweden Joint VT-COM-IT Chapter in 2014. He is an Area Editor of *Computer Networks*.