Roadside Unit Caching: Auction-Based Storage Allocation for Multiple Content Providers

Zhiwen Hu, *Student Member, IEEE*, Zijie Zheng, *Student Member, IEEE*, Tao Wang, *Senior Member, IEEE*, Lingyang Song, *Senior Member, IEEE*, and Xiaoming Li, *Senior Member, IEEE*

***Abstract*— Recent improvements in vehicular ad hoc networks are accelerating the realization of intelligent transportation system (ITS), which not only provides road safety and driving efficiency, but also enables infotainment services. Since data dissemination plays an important part in ITS, recent studies have found caching as a promising way to promote the efficiency of data dissemination against rapid variation of network topology. In this paper, we focus on the scenario of roadside unit (RSU) caching, where multiple content providers (CPs) aim to improve the data dissemination of their own contents by utilizing the storages of RSUs. To deal with the competition among multiple CPs for limited caching facilities, we propose a multi-object auction-based solution, which is sub-optimal and efficient to be carried out. A caching-specific handoff decision mechanism is also adopted to take advantages of the overlap of RSUs. Simulation results show that our solution leads to a satisfactory outcome.**

***Index Terms*— Vehicular networking, roadside unit, caching, multi-object auction.**

I. Introduction

VEHICULAR ad hoc networks (VANETs) are becoming increasingly popular in recent years, aiming to cope with the strong demands for communicating on the move. As more and more communication and computing techniques being enabled by VANETs, it is promising to deploy Intelligent Transportation Systems (ITS) widely in our real world [2]. By combining the theoretical improvements with the development of transportation infrastructure, ITS is expected to alleviate or even prevent many road traffic problems such as congestions and accidents effectively. To achieve these targets, roadside units (RSUs) are being deployed as the most significant infrastructure in ITS [3]. RSUs are typically Internet-connected devices, dedicated in exchanging information with on-board units (OBUs) placed at vehicles. Therefore, vehicle-to-roadside (V2R) communications are enabled in addition to the vehicle-to-vehicle (V2V) communications.

Manuscript received September 6, 2016; revised January 3, 2017 and March 31, 2017; accepted June 15, 2017. Date of publication July 11, 2017; date of current version October 9, 2017. This work was supported in part by the National 973 Project under Grant 2013CB336700 and in part by the National Nature Science Foundation of China under Grant U1301255 and Grant 61625101. This paper was presented at the ACM International Symposium on Mobile Ad Hoc Networking and Computing, Hangzhou, China, June 2015 [1]. The associate editor coordinating the review of this paper and approving it for publication was P. Wang. *(Corresponding author: Lingyang Song.)*

The authors are with the School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China (e-mail: zhiwen.hu@pku.edu.cn; zijie.zheng@pku.edu.cn; wangtao@pku.edu.cn; lingyang.song@pku.edu.cn; lxm@pku.edu.cn).

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

Digital Object Identifier 10.1109/TWC.2017.2721938

Although initially designed to improve road safety and driving efficiency, ITS can also provide infotainment services for the passing-by drivers and passengers with the help of RSUs, such as commercial, informative, and entertainment services [4]. One of the prerequisites for infotainment services is to design the data dissemination strategy in VANET environment, where data can either be generated by the OBUs in VANETs or by the content provider (CP) on the Internet. In both cases, wireless data need to be disseminated to the given set of target vehicular users through VANET. However, due to the rapid changes in network topology and high variability of the connectivity, it is hard to guarantee that data can arrive at targets safely, accurately and punctually [5]. Therefore, data dissemination in VANET still remains to be a challenging task.

To deal with this problem, many early studies considered cache-enabled RSUs and OBUs as the ways to improve the quality of dissemination [6]–[16]. Among them, [6]–[9] studied the case where data was generated by OBUs in VANET and being disseminated to other interested users. While [10]–[16] focused on the case where data was provided by CPs and to be disseminated to interested users VANETs. Specifically, Abdelhamid *et al.* [6] and [7] suggested that cache-enabled RSUs can collect OBU-generated data in order to serve other OBUs. The works in [8] and [9] further took into account the caching ability of OBUs and thus made the dissemination more efficient. To accelerate the speed of vehicular users acquiring online contents of CPs, centralized RSU caching algorithms were proposed by Ding *et al.* [10] and Idir *et al.* [11]. User-specific RSU caching was discussed in [12]–[14], where CPs' contents can be cached in the RSUs on the pre-determined routes of vehicular users. Works in [15] and [16] also considered the caching ability of OBUs as a supplement to cache-enabled RSUs. In addition, coding techniques are discussed in [17]–[19], which provide efficient ways to utilize caching storages.

For the case where data are generated by CPs, however, few existing studies consider the existence of multiple CPs [20]. In reality, there are multiple CPs possessing different sets of contents, and vehicular users are able to select their interested contents to download through VANETs. Since each CP only cares about the experience of its own users, it intends to improve the data disseminations of its own contents by utilizing the caching storages of RSUs or OBUs. Unfortunately, the storage capacity of either RSUs or OBUs is barely enough for any of the CPs, which makes the competition among different CPs unavoidable. Therefore, a proper mechanism should be designed to deal with the competition

among CPs and guarantee the overall performance at the same time.

Since the capacity of RSUs can be much higher than that of OBUs and most of the users' retrieved data result from V2R communications rather than V2V communications [21], the major contribution for the dissemination of CPs' data is RSU caching. Therefore, without the loss of generality, in this paper, we focus on RSU caching for multiple CPs and formulate this problem by taking into account the overlapping among RSUs. Since CPs have to compete for the caching storages on behalf of their own contents, a nature solution is to apply auctions to allocate resources reasonably [22]–[24]. Each CP has to evaluate its contents and bid for caching storages, while the Mobile Network Operators (MNOs) who manages the RSUs can benefit from CPs' payments [25]. We propose our own solution based on multi-object auctions, where a serial of multi-object auctions are carried out to complete the caching scheme. Each multi-object auction can be solved by the market matching algorithm [36], which uses the valuations to calculate allocation results and trading prices. Simulation results show the effectiveness of the proposed solution and also testified our theoretical predictions.

The main contributions of our work are listed below:

1) We focus on the roadside unit caching scenario which involves multiple CPs that are competing for the limited caching storages of RSUs.
2) We formulate the caching problem with the objective to maximize the total amount of downloaded data, where a caching-specific handoff mechanism are adopted due to the overlap of RSUs.
3) We provide a sub-optimal solution based on multi-object auctions, which is efficient to be carried out and also compatible with the existence of multiple MNOs.
4) We testify the effectiveness of our auction-based solution, the caching-specific handoff mechanism, and the proposed content block segmentation method by simulations.

The rest of our paper is organized as follows. Section II presents our system model and problem formulation of roadside caching. Section III provides the theoretical analysis on the system parameters. Section IV introduces our multi-object auction based solution. Section V shows the simulation results which prove the effectiveness of our solution and testify our theoretical analysis. Finally, we conclude our paper in Section VI.

II. System Model and Problem Formulation

As shown in Fig. 1, we consider a simple two-way street where RSUs with caching abilities are equidistantly distributed.[1] Vehicular users that moving in both directions can get accesses to nearby RSUs and thereby download contents from the Internet. We assume that some of the users start to download one of the contents once they drive into the street. During the whole journey, users can always continue downloading if there are RSUs nearby. But the transmission rate could be different, depending on whether the desired content is cached in nearby RSUs. Since most of the retrieved data of vehicular users result from V2R communications [21], we ignore the possible V2V communications for simplicity. As the downloading process may not complete by the end of the street, we aim to maximize the total amount of downloaded data by caching the contents properly. The replacement procedure of caching storages can be done during the night, when RSUs are lightly loaded. For the rest part of this section, we provide the model of contents, the model of vehicles, the model of RSUs, the model of wireless communications, the adopted handoff decision, and finally the objective function.

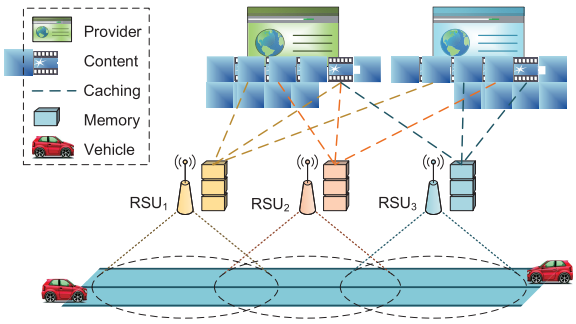


Fig. 1. System model for roadside unit caching among multiple content providers.

A. Contents

There are L CPs that aim to provide vehicular users with better quality of service by caching their individual contents into RSUs. We denote these CPs by $\mathcal{P} = \{P_l | 1 \leq l \leq L\}$, and denote the k^{th} content of P_l by $C_{l,k}$, with size $S_{l,k}$. The popularity of $C_{l,k}$ is its probability of being requested by each single user, denoted by $\phi_{l,k}$. It is assumed that $\phi_{l,k}$ can be seen as a constant in a single day, and is also predictable with the help of some learning techniques [26]. The popularity distribution of massive amount of contents conform to Zipf-like distribution [27], which is kind of power law distribution. In addition, we also have $\sum_{l,k} \phi_{l,k} = 1$, to guarantee that each interested user only choose one content to download. Since contents can be divided into several segments for caching, we use $C_{l,k,n}$ to denote the n^{th} segment of $C_{l,k}$, and $S_{l,k,n}$ to denote the size of it, where $1 \leq n \leq N_{l,k}$, and $N_{l,k}$ is the number of segments that $C_{l,k}$ is divided. Here we assume that different segments of a certain content have a sequential order, and users have to download the segments of their desired contents in sequence.[2]

B. Vehicles

The traffic condition on the street can be described by three parameters, which are the linear density of vehicles $\rho(x,t)$, the velocity of vehicles $u(x,t)$, and the flux of vehicles

[1] This model can be generalized for a complicated traffic network as long as the traffic condition can be estimated.

[2] Although a coding method is also applicable (which ignores the sequence of segments), the method we adopted here is compatible with streaming media (which users would like to download in sequence).

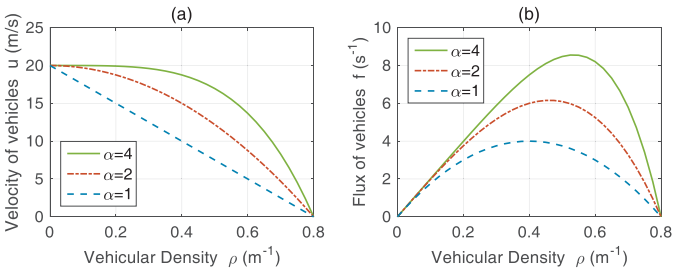


Fig. 2. Generalized LWR vehicular traffic model, where $\rho_{max} = 0.8m^{-1}$ and $u_{max} = 20m/s$, with $\alpha = 1, 2, 4$ respectively.

$f(x,t) = \rho(x,t)v(x,t)$. And we also adopt the generalized Lighthill-Whitham-Richards (LWR) model [28] to describe the additional restriction between velocity and density, given as:

$$u(\rho) = u_{max}\left[1 - \left(\frac{\rho}{\rho_{max}}\right)^{\alpha}\right], \quad \alpha = 1, 2, 3 \cdots \tag{1}$$

where $u_{max} > 0$ is the maximum allowable velocity in the given street, and ρ_{max} is the maximum allowable vehicular density. Parameter α can be called as traffic coefficient, which describes the property of vehicular traffic: A greater α indicates that the vehicular velocity is not critically influenced by vehicular density until the street is crowded. When $\rho = \rho_{max}$, the u decreases to zero, indicating that the street is congested. An illustration of this model is shown in Fig. 2.

During a day, the vehicular traffic condition of the street could vary drastically. However, we suppose that the traffic condition on each side of the street is uniform in a certain period of time. That is to say, ρ, u and f are no longer the functions of location x within each time slot. Let t denote a certain time slot and let T denote the total number of time slots in a day. Later in this paper, we use ρ_1^t, u_1^t and f_1^t respectively to represent the traffic from the left side of the street in the t^{th} time slot, and use ρ_2^t, u_2^t and f_2^t to represent the other side. Among all the users, only a proportion of them is interested in downloading contents as driving through the street, while the others don't demand this service and only contribute to vehicular traffic. And we denote such proportion as λ, where $0 \leq \lambda \leq 1$.

C. RSUs

There are M RSUs equidistantly distributed along the street, denoted by $\{RSU_m\}$, where $1 \leq m \leq M$. These RSUs can belong to multiple MNOs, and the storage capacity of RSUs could be different. We use S_m to denote the storage capacity of RSU_m. The coverage length of any RSU is assumed to be the same, denoted by d, while the total length of the street is denoted by d_s. Any user that wants to connect to a specific RSU has to make sure that the distance between its vehicle and the RSU is less than $d/2$. In addition, users in the overlapping region of the given two RSUs are able to benefit from both RSUs' caching storages. Also note that the width of the street is ignored for simplicity.

Since the storages of RSUs can be used to cache contents for users, we use the boolean matrix Δ to represent the caching of RSUs, whose element is defined as:

$$\delta_{l,k,n}^{m} = \begin{cases} 1, & \text{if } \text{RSU}_m \text{ caches } C_{l,k,n}, \\ 0, & \text{if } \text{RSU}_m \text{ doesn't cache } C_{l,k,n}. \end{cases} \tag{2}$$

D. Wireless Traffic

There are two types of wireless connections with transmission constraints in our model, which are the down-links from RSUs to vehicles and the backhaul links from RSUs to the Internet.

As for the down-links, each RSU is assumed to have U wireless channels and the maximum rate of each channel is R_{down}. For simplicity, we suggest that each user can only occupy at most one channel at a time, and multiple users are able to share a channel by time-division multiplexing. Based on the assumptions on vehicles, the average number of users that intend to connect to the Internet under each RSU's coverage is $d(\rho_1^t + \rho_2^t)\lambda$. If $d(\rho_1^t + \rho_2^t)\lambda \leq U$, then the down-link rate for each user connected to a RSU is R_{down}. Otherwise, the down-link rate is $\frac{R_{down}U}{d(\rho_1^t + \rho_2^t)\lambda}$. By combining these two situations, we provide the following equation[3]:

$$r_{down}^{t} = \min\left\{R_{down}, \frac{R_{down}U}{d(\rho_1^t + \rho_2^t)\lambda}\right\}. \tag{3}$$

As for the backhaul links, we denote the maximum backhaul rate of a RSU as R_{back}. Here we have to further introduce the cache failure ratio of RSU_m, μ_m, where $0 \leq \mu_m \leq 1$, to denote the percentage of data that cannot be served by the caching storage of RSU_m. Therefore, the equivalent number of users that occupy the backhaul link of RSU_m is $d(\rho_1^t + \rho_2^t)\lambda\mu_m$. Similar to the downlink rate, we can provide the expression of the available backhaul link rate for each user as follows:

$$r_{back}^{m,t} = \min\left\{R_{back}, \quad \frac{R_{back}}{d(\rho_1^t + \rho_2^t)\lambda\mu_m}\right\}. \tag{4}$$

Based on the equations above, we can provide the final download rate of each user as below:

$$r_{cache}^{m,t} = r_{down}^{m,t}, \tag{5}$$

$$r_{uncache}^{m,t} = \min\{r_{down}^{m,t}, r_{back}^{m,t}\}, \tag{6}$$

where $r_{cache}^{m,t}$ is for the users whose desired content is cached in RSU_m at t, and $r_{uncache}^{m,t}$ is for the users whose desired content is not cached in RSU_m at t.

E. Handoff Decision

When a vehicle moves from one RSU to another adjacent RSU, a handoff process should be considered, which usually includes network discovery, handoff decision and handoff execution [29]. The network discovery phase is to find adjacent connectable RSUs for users, which can be easily handled by a RSU controller [30] since the adjacency relationship between

[3]Note that, for the user that is in the overlapping region of any two RSUs, whichever RSU it is connected to, it causes interference to the other RSU in the same channel and affects both RSUs' download speed. Therefore it is reasonable to assume that the rate of a RSU depends on the number of all the vehicles under its cover (with a λ factor).

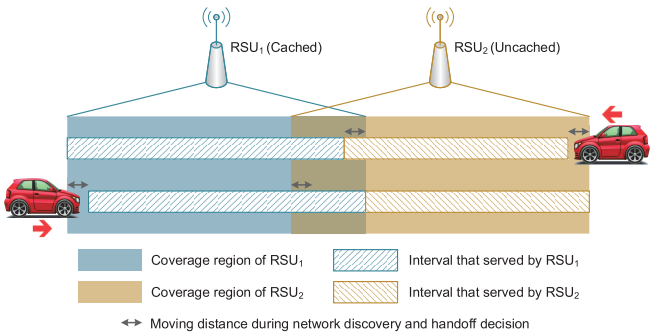


Fig. 3. An example of the adopted handoff decision. RSU_1 has cached a specific content while RSU_2 does not. The car from right makes handoff execution immediately, while the car from left makes handoff execution after leaving RSU_1.

RSUs is quite simple in our model. In addition, the delay of handoff execution can be neglected since most time is spent on the previous two phases. However, the handoff decision should be better designed to improve the quality of caching service in our model. We assume that, if a user is moving towards a RSU that does not cache the desired content segment, the handoff execution will not be made hurriedly until the user has left the coverage region of the previous RSU. In comparison, if a user is moving towards a RSU that caches the desired content segment, the handoff execution will be made as soon as possible. The download rate of each user can be improved by selecting the cached RSU as priority during the handoff decision.

We call this handoff decision as *caching-specific handoff*, which can be applicable in practice since a RSU controller is aware of whether a user's desired content is cached. Here, we use t_h to represent the total delay of network discovery and handoff decision. It is only after having entered the coverage region for t_h that a user can set up connections to a certain RSU. According to the adopted handoff decision, whether a user immediately connects to a RSU after t_h depends on whether the desired content is cached. Fig. 3 gives an example of such handoff decision.

F. Amount of Downloaded Data

Once a user with interests in $C_{l,k}$ enters the street from the left side at t, it has to download $N_{l,k}$ segments of $C_{l,k}$ sequentially. By the end of the street, the amount of downloaded data is denoted by $D^t_{1,l,k}$, where $D^t_{1,l,k} \leq S_{l,k}$. Similarly, for the other direction, the amount of downloaded data is $D^t_{2,l,k}$, where $D^t_{2,l,k} \leq S_{l,k}$. Given the vehicular condition, the caching allocation and the handoff delay, $D^t_{1,l,k}$ and $D^t_{2,l,k}$ can be calculated. The general mathematical expression of $D^t_{1,l,k}$ and $D^t_{2,l,k}$ has a complicated form since $N_{l,k}$ could be greater than 1. To keep the model more readable, we put the detailed calculation procedure in Section III. Our objective is to maximize the expected total amount of downloaded data within each day, as given below:

$$\max_{\Delta} \sum_{t=1}^{T} \left\{ \lambda f_1^t \sum_{l,k} D^t_{1,l,k} \cdot \phi_{l,k} + \lambda f_2^t \sum_{l,k} D^t_{2,l,k} \cdot \phi_{l,k} \right\} \quad (7)$$

$$s.t. \sum_{l,k,n} \delta^m_{l,k,n} \cdot S_{l,k,n} \leq S_m, \quad \forall m, \quad (8)$$

where the constraint shows the limited storage capacity of each RSU.

III. Theoretical Analysis

In this section, we first discuss the calculation of downloaded data of each user, then reveal some properties of the formulated problem, and finally analyse the impacts of some parameters on the caching performance.

A. Calculation of Downloaded Data

For a specific content segment $C_{l,k,n}$, given its allocation in M RSUs and the traffic condition in the street, we can calculate the amount of data downloaded by the users who request $C_{l,k,n}$. As given in Section II, u^t_1 denotes the vehicular velocity from the left side of the street. Thus the travel distance induced by handoff delay for the users to connect to a RSU is $d^t_{h,1} = t_h \cdot u^t_1$. Based on the caching result of $C_{l,k,n}$, the whole street can be divided into M *serving intervals*, where the m^{th} interval represents the scope where RSU_m transmits $C_{l,k,n}$ to users moving in the given direction. For the traffic from the left, we denote $P^{m,t}_{1,l,k,n}$ and $Q^{m,t}_{1,l,k,n}$ as the beginning coordinate and the ending coordinate of such an interval. The overlapping distance of two adjacent RSUs is given by d_o and the location of RSU_m is given by x_m, where RSU_1 is the leftmost one and RSU_M is the rightmost one. Based on the adopted handoff decision, we have

$$P^{m,t}_{1,l,k,n} = \begin{cases} x_m - d/2 + d^t_{h,1}, & \text{if } C1, \\ x_m - d/2 + \max\{d^t_{h,1}, d_o\}, & \text{if } C2. \end{cases} \quad (9)$$

$$Q^{m,t}_{1,l,k,n} = \begin{cases} x_m + d/2 + \min\{0, d^t_{h,1} - d_o\}, & \text{if } C3, \\ x_m + d/2, & \text{if } C4. \end{cases}$$

$$C1\text{: } m = 1 \text{ or } \delta^m_{l,k,n} = 1; \quad C2\text{: } m > 1 \text{ and } \delta^m_{l,k,n} = 0;$$
$$C3\text{: } m < M \text{ and } \delta^{m+1}_{l,k,n} = 1; \quad C4\text{: } m = M \text{ or } \delta^{m+1}_{l,k,n} = 0. \quad (10)$$

The upper line in Eqn. (9) represents an immediate entering since the RSU has cached the content or it is the first RSU being encountered. The lower line in Eqn. (9) represents a deferred entering since the content is cached in the previous RSU and not cached in the current RSU. The upper line in Eqn. (10) shows an immediate leaving since the next RSU has cached the content. And the lower line in Eqn. (10) shows a deferred leaving since the next RSU has not cached the content or there is no RSUs ahead. Note that if $d^t_{h,1} \leq d_o$, we have $Q^{m,t}_{1,l,k,n} = P^{m+1,t}_{1,l,k,n}$, thus the downloading process is continuous. However, if $d^t_{h,1} > d_o$, we have $Q^{m,t}_{1,l,k,n} < P^{m+1,t}_{1,l,k,n}$, which implies there are some regions where users are off the connection.

The maximum amount of downloaded data that a user can acquire while driving through the interval of RSU_m is $r \cdot (Q^{m,t}_{1,l,k,n} - P^{m+1,t}_{1,l,k,n})/v^t_1$, where $r = r^{m,t}_{cached}$ if $\delta^m_{l,k,n} = 1$ and $r = r^{m,t}_{uncached}$ if not. With the consideration of $N_{l,k}$ (the number of segments that $C_{l,k}$ is divided into), the user that interested in $C_{l,k}$ has to first download $C_{l,k,1}$ according to the intervals of $C_{l,k,1}$. After $C_{l,k,1}$ has been successfully received, the user continues to download $C_{l,k,2}$ in the current

location according to the intervals of $C_{l,k,2}$ (if $C_{l,k,2}$ exists). This process goes on until all the segments of $C_{l,k}$ have been downloaded or the user leaves the street with only a fraction of $C_{l,k}$ downloaded.

In comparison, the traditional handoff decision is based on the Received Signal Strength (RSS) measurement [31], where an OBU is likely to switch to a new RSU near the middle of two RSUs[4] regardless of whether the content is cached. The intervals of a traditional handoff decision can be given by

$$P^{m,t\prime}_{1,l,k,n} = \begin{cases} x_m - d/2 + d^t_{h,1}, & \text{if } m = 1 \\ x_m - d/2 + \max\{d^t_{h,1}, d_o/2\}, & \text{if } m > 1. \end{cases} \quad (11)$$

$$Q^{m,t'}_{1,l,k,n} = \begin{cases} x_m + d/2 - d_o + \max\{d^t_{h,1}, d_o/2\}, & \text{if } m < M, \\ x_m + d/2, & \text{if } m = M \end{cases} \quad (12)$$

By subtracting the intervals of the traditional handoff and that of the caching specific handoff, it is not hard to find that $Q^{m,t}_{1,l,k,n} - P^{m,t}_{1,l,k,n} \geq Q^{m,t\,\prime}_{1,l,k,n} - P^{m,t\,\prime}_{1,l,k,n}$ if $C_{l,k,n}$ is cached in RSU_m and $Q^{m,t}_{1,l,k,n} - P^{m,t}_{1,l,k,n} \leq Q^{m,t\,\prime}_{1,l,k,n} - P^{m,t\,\prime}_{1,l,k,n}$ if not. Such property guarantees that the caching specific handoff provides users with longer scope to download with higher rate. And we have proved this conclusion in Section V-B by simulations.

For the vehicles from the right side of the street, the calculation of downloaded data is similar to that of the above discussion. Therefore, we only provide the corresponding expressions of the intervals and omit the detailed discussions and comparisons:

$$P^{m,t}_{2,l,k,n} = \begin{cases} x_m + d/2 - d^t_{h,1}, & \text{if } C1, \\ x_m + d/2 - \max\{d^t_{h,1}, d_o\}, & \text{if } C2. \end{cases}$$

$$Q^{m,t}_{2,l,k,n} = \begin{cases} x_m - d/2, & \text{if } C3, \\ x_m - d/2 - \min\{0, d^t_{h,1} - d_o\}, & \text{if } C4. \end{cases}$$

$C1$: $m = M$ or $\delta^m_{l,k,n} = 1$; $C2$: $m < M$ and $\delta^m_{l,k,n} = 0$;
$C3$: $m = 1$ or $\delta^{m-1}_{l,k,n} = 0$; $C4$: $m > 1$ and $\delta^{m-1}_{l,k,n} = 1$.

B. Two Properties of the Formulated Problem

In this subsection, we prove that our problem is: NP-hard [32] and submodular [33].

Proposition 1: The problem formulated in the Eqn. (7) is NP-hard.

Proof: To achieve this, we prove that a special case of our problem can be reduced to the 0-1 Knapsack Problem [34] in polynomial time. We define the special case of our problem as:

Single RSU Caching Problem: A RSU with limited storage capacity S is located in a street with vehicular density ρ and zero handoff delay. The contents $\{C_k\}$ with corresponding size $\{S_k\}$ are requested by vehicular users according to their popularity $\{P_k\}$. Assuming that the rate of backhaul link r_{back} and the downlink rate r_{down} are fixed for each user ($r_{back} < r_{down}$), find the optimal caching strategy to maximize the average downloaded data.

To deal this problem, we can calculate the caching gain of each content, denoted as G_k, representing the marginal increase of average downloaded data as we cache C_k into the RSU. If we reckon each content as an object, then there is a value G_k and a cost S_k of caching it. Therefore, the problem can be considered as a knapsack problem without further transformation. Since a knapsack problem is NP-hard and the reduction only takes polynomial time, we can deduce that the *Single RSU Caching Problem* is NP-hard. Therefore, the more general and complicated problem in (7) is also NP-hard. ■

Before we prove the submodularity, we have to provide some notations. We first use $\mathcal{A}$ to denote a given caching allocation, which includes M groups of contents being cached into the M RSUs. And we denote $f : \mathcal{A} \to \mathbb{R}$ as the function that maps the given caching allocation to a real number which represents the system performance (i.e., the total amount of downloaded data). By putting some additional contents into $\mathcal{A}$, we have another caching allocation $\mathcal{B}$, where $\mathcal{A} \subseteq \mathcal{B}$. The function f is a submodular function if and only if

$$f(\mathcal{A} \cup x) - f(\mathcal{A}) \geq f(\mathcal{B} \cup x) - f(\mathcal{B}), \quad \mathcal{A} \subseteq \mathcal{B}, \quad (13)$$

where x denotes the single operation of caching one of the contents into one of the RSUs. Intuitively, this attribute implies that the marginal utility is monotonically decreasing (adding x into $\mathcal{A}$ could be better than adding x into a bigger set $\mathcal{B}$).

Proposition 2: The problem formulated in Eqn. (7) is submodular.

Proof: Assuming that the current caching allocation is not completed, we consider the effect of caching another content into a certain RSU: 1) If it is already cached in this RSU, then no more gains can be acquired. 2)If it is not cached in this RSU, then there are two parts of potential marginal gains. The first one G_{This}, is for the users who request this content, which is either zero or positive (depending on whether this content can be already finished downloading for the drive-through users based on its size). The second one, G_{All}, is for all of the users (also either positive or zero), since caching can reduce the cache failure ratio of the RSU, which could increase the backhaul rate according to Eqn. (4). All the above considerations show that marginal gain of caching is non-negative.

For the case where $\mathcal{A} = \mathcal{B}$, we have $f(\mathcal{A} \cup x) - f(\mathcal{A}) = f(\mathcal{B} \cup x) - f(\mathcal{B})$ and the above property holds. For the case where $\mathcal{A} \neq \mathcal{B}$, we denote $\mathcal{C} = \mathcal{B} - \mathcal{A}$ as the additional contents that added to the allocation $\mathcal{A}$. If $x \subseteq \mathcal{A}$, then $f(\mathcal{A} \cup x) - f(\mathcal{A}) = f(\mathcal{B} \cup x) - f(\mathcal{B}) = 0$. If $x \subseteq \mathcal{C}$, we have $f(\mathcal{A} \cup x) - f(\mathcal{A}) \geq 0 = f(\mathcal{B} \cup x) - f(\mathcal{B})$. And finally, if x is not in $\mathcal{B}$, we can also obtain $f(\mathcal{A} \cup x) - f(\mathcal{A}) \geq f(\mathcal{B} \cup x) - f(\mathcal{B})$ since the presence of $\mathcal{C}$ might reduce G_{This} and G_{All} of x. Such a reduction occurs when the addition of $\mathcal{C}$ makes vehicular users able to completely download some of the contents. In conclusion, we always have $f(\mathcal{A} \cup x) - f(\mathcal{A}) \geq f(\mathcal{B} \cup x) - f(\mathcal{B})$. Therefore, the formulated problem is submodular. ■

[4] A more realistic model will consider the signal strength, and the handoff is executed later than the middle of two RSUs. However, there is one property that remains the same in our formulation: RSUs are expected to serve for the same distance and the difference from (11) (12) is not significant.

Due to its submodularity, in the rest of our paper we are able to compare our auction solution with a general centralized algorithm of submodular problems. Briefly, this centralized algorithm select the most valuable allocation pair (one content to one RSU) within each step and repeats until there is not enough caching space. It is guaranteed that the outcome is at least $(1-1/e)$ of the optimal solution [33]. Therefore, its outcome can be taken as a benchmark in our performance evaluation in Section V-B.

C. Impacts of System Parameters

In this subsection, we analyse some of the parameters which can affect the total amount of downloaded data. Since vehicular density ρ is the independent variable of velocity and flux, and also influences download rate of users, it is necessary to discuss the its influence on each time slot's performance. Besides, the competition for limited caching storages among CPs' contents is the key issue, thus the total number of contents K and the storage capacity of RSUs S are the two important concerns. In addition, the number of RSUs M can also affect the outcome because it determines the degree of overlapping as the length of the street is fixed. At last, the proportion of interested users λ and the handoff delay t_h are also two influential factors that need to be discussed.

To have a better understanding of the problem, in the rest of this subsection we consider a special case where the two sides of the street share the same traffic condition, i.e., $\rho_1^t = \rho_2^t$, $u_1^t = u_2^t$ and $f_1^t = f_2^t$. Note that the objective function (7) shows that the total amount of data is the sum of each time slot's downloaded data, and given a fixed caching allocation the outcomes of different time slots are independent of each other. Therefore, we mainly analyse how the parameters changes the amount of downloaded data in each time slot. And we use ρ, u and f to denote the density, velocity and flux of the street from both directions in a certain time slot.

Proposition 3: With a fixed caching allocation, given that all contents are cached without being segmented, the amount of downloaded data that takes ***vehicular density*** ρ *as a single variable is concave within* $(0, \rho_{max})$.

Proof: Given the vehicular density ρ, we have vehicular velocity $u = u_{max}(1-\rho/\rho_{max})$ and vehicular flux $f = u_{max}(1-\rho/\rho_{max})\rho$. Given the maximum backhaul rate R_b and maximum down-link rate R_d, we have the cached rate of RSU_m for each user given by $r_c^m = \min\left\{R_d, \frac{R_d U}{2\lambda\rho}\right\}$ and the uncached rate by $r_u^m = \min\left\{R_d, \frac{R_d U}{2d\lambda\rho}, R_b, \frac{R_b}{2d\lambda\mu_m\rho}\right\}$.

Since the allocation is fixed, RSU_m's cache failure ratio μ_m can be seen as a constant. In addition, the serving intervals that introduced in the last subsection are also fixed. We denote the distance of RSU_m's serving interval as d_m, thus the maximum amount of data that a user can acquire as passing by RSU_m is $D_{l,k}^m = r^m\frac{d_m}{u} = \frac{r^m\cdot d_m}{u_{max}(1-\rho/\rho_{max})}$, where $r^m = r_c^m$ if RSU_m caches $C_{l,k}$ and $r^m = r_u^m$ if not. When a user is interested in $C_{l,k}$, the amount of downloaded data by the end of the street is $D_{l,k} = \min\left\{\sum_m D_{l,k}^m,\ S_{l,k}\right\}$. The expected total amount of data that downloaded from both directions of the street is

$$\begin{aligned} D &= 2\lambda f\sum_{l,k}\phi_{l,k}D_{l,k} \\ &= 2\sum_{l,k}\left[\phi_{l,k}\min\left\{\sum_m \lambda f D_{l,k}^m \lambda f S_{l,k}\right\}\right] \\ &= 2\sum_{l,k}\left[\phi_{l,k}\min\left\{\sum_m \frac{\lambda f\cdot r^m d_m}{u_{max}(1-\frac{\rho}{\rho_{max}})}, \lambda f S_{l,k}\right\}\right] \\ &= 2\sum_{l,k}\left[\phi_{l,k}\min\left\{\sum_m \lambda\rho\cdot r^m d_m \lambda f S_{l,k}\right\}\right] \end{aligned} \tag{14}$$

For all of the cached contents, $\lambda\rho\cdot r^m = \min\left\{\lambda\rho R_d, \frac{R_d U}{2}\right\}$. And for all of the uncached contents, we have $\lambda\rho\cdot r^m = \min\left\{\lambda\rho R_d, \frac{R_d U}{2d}, \lambda\rho R_b, \frac{R_b}{2d\mu_m}\right\}$. For either case, $\lambda\rho\cdot r^m$ is concave of ρ [35]. In addition, $f = u_{max}(1-\rho/\rho_{max})\rho$ is also concave of ρ, which makes $\min\left\{\sum_m \lambda\rho r^m d_m, \lambda f S_{l,k}\right\}$ concave. Finally, we can deduce that D is concave of ρ. ■

Remark 1: When the vehicles is sparse (such as in the midnight) or crowded (such as in the busy hours), the efficiency of the data services provided for users will not be satisfying. The former case is due to the lack of users in the street, while the latter case is due to the fact that crowded vehicles move slowly and thereby reduce the vehicular flux. However, a moderate vehicular density (such as in the normal hours) can make the best use of RSUs' caching.

Proposition 4: Supposing that the storage capacities of RSUs are the same, then the total amount of downloaded data D has a positive correlation with ***storage capacity*** *S.*

The proof of this proposition is trivial and the conclusion is straight-forward. The key point is to notice that the caching allocation of S' can be derived from the given caching allocation of S $(S < S')$. Therefore additional content segments can be added to the caching storages and the involved contents are able to be downloaded more by users.

Proposition 5: Given a certain distribution of content popularity and content size, the total amount of downloaded data D has a negative correlation with ***the number of contents*** *K.*

Proof: Suppose that there are initially K contents in the system and we denote the set of these content as $\mathcal{C}$. The downloaded data in a specific time slot can be calculated by

$$D = \sum_{C_{l,k}\in\mathcal{C}}\phi_{l,k}D_{l,k},$$

where $D_{l,k} = \lambda f_1 D_{1,l,k} + \lambda f_2 D_{2,l,k}$. Now we add additional set of contents $\mathcal{C}'$ with the same popularity distribution and size distribution, supposing $|\mathcal{C}'| = K' = xK$ and $x > 0$. Then the downloaded data of each original content in $\mathcal{C}$ is expected to be lowered since some of the caching storages are now occupied by new contents, i.e., $D'_{l,k} < D_{l,k}$. Besides, the popularity of all the original contents and all the new contents decreases because the popularity should be normalized, as: $\phi'_{l,k} = \phi_{l,k}\frac{1}{1+x}$, if $C_{l,k}\in\mathcal{C}$. Note that no matter from $\mathcal{C}$ or $\mathcal{C}'$, contents with the similar popularity and size are expected to have similar amount of downloaded data for users (as long

as the caching allocation algorithm is rational). Therefore, the downloaded data in a specific time slot with $\mathcal{C}'$ added is

$$\begin{aligned} D' &= \sum_{C_{l,k}\in\mathcal{C}} \phi'_{l,k} D'_{l,k} + \sum_{C_{l,k}\in\mathcal{C}'} \phi'_{l,k} D'_{l,k} \\ &= \left(1+x\right) \sum_{C_{l,k}\in\mathcal{C}} \phi'_{l,k} D'_{l,k} \\ &= \sum_{C_{l,k}\in\mathcal{C}} \phi_{l,k} D'_{l,k} < \sum_{C_{l,k}\in\mathcal{C}} \phi_{l,k} D_{l,k} = D. \end{aligned}$$

Therefore, the increase in content number K reduces the amount of downloaded data D. ■

This result can also be intuitively comprehended that the increase in content number leads to the decrease in caching percentage, making the caching system less efficient.

Proposition 6: Given that the length of the street is constant and the RSU are distributed equidistantly, the total amount of downloaded data D has a positive correlation with the ***number of RSUs*** *M.*

Note that the download rate is not influenced by the number of RSUs (as modeled in section II and explained in the corresponding footnote). Since deploying more RSUs enables more contents to be cached, the amount of downloaded data of each content for users is also expected to be larger.

Proposition 7: With the contents unsegmented, the total amount of downloaded data D has a positive correlation with the ***proportion of interested users*** *λ.*

Proof: Based on the denotations used in Proposition 3, $\lambda\rho \cdot r^m$ is either $\min\left\{\lambda\rho R_d, \frac{R_d U}{2}\right\}$ or $\min\left\{\lambda\rho R_d, \frac{R_d U}{2d}, \lambda\rho R_b, \frac{R_b}{2d\mu_m}\right\}$. In both cases, $\lambda\rho \cdot r^m$ has a positive correlation with λ. Based on Eqn. (14), which tells us the positive correlation of $\lambda\rho \cdot r^m$ and D, we can directly obtain that D has a positive correlation with λ. ■

Proposition 8: With fixed setting of RSUs, contents, traffic condition, caching allocation, the total amount of downloaded data D has a negative correlation with ***handoff delay*** *t_h .*

This conclusion is also straight-forward to intuitively understand. The impact of t_h mainly lies in the distance induced by handoff delay, $d_{h,1}$ and $d_{h,2}$. And the increase of $d_{h,1}$ and $d_{h,2}$ defers the starting point of connecting to a cached RSU. Thus each user is expected to download less data while driving through the street.

IV. Auction-Based Solution

In this section, we propose an auction-based solution to deal with the caching problem which involves multiple CPs. In general, CPs who possess different sets of contents play the roles of bidders, and the RSUs' caching storages that owned by MNOs play the roles of objects. A serial of multi-object auctions are hold to determine the caching allocation for the next day. And based on the auction results, old cached contents are replaced by new ones during midnight when RSUs are lightly loaded.

For the rest part of this section, we first discuss the motivation of applying multi-object auction, then elaborate the setup of a serial of multi-object auctions, and finally introduce the market matching algorithm to solve each auction.

A. Background and Motivation

The RSU caching problem we are dealing with involves multiple selfish CPs. Each CP aims to cache its own contents into RSUs in order to provide better experience for its own users and thereby acquires better user ratings. However, the caching storages of RSUs are not infinite, and hardly sufficient for caching enough contents to satisfy all kinds of requests from users. Therefore, the competition among CPs is inevitable. Additionally, MNOs, the owners of RSUs, have to pay for the installation and operation costs of caching storages in RSUs. Thus they are not interested to maintain the storages to cache the contents of CPs unless they can benefit from it. Since CPs are willing to pay a reasonable price to obtain caching storages, it is possible to create transactions between CPs and MNOs. For each CP, there is a value of each content being cached, which, however, the MNOs are not aware of. Therefore auctions can be applied here, to determine the trading prices according to the bids submitted by CPs. A proper auction mechanism [22] is able to guarantee caching efficiency as well as providing fairness for all the participated CPs.

In our situation, the auctioneer of the auctions (who is in charge of the auction procedure) could be another third party, or even be the MNO itself as long as there is only one MNO. The objects to be sold in each auction are the caching storages of RSUs, which can be seen as successive in size. The complexity of auctioning storages in random sizes is extremely high and almost intractable in practice. Thus, it is reasonable to divide the caching storages into specific equal-sized storage blocks, and regard each of blocks as a single object in the auctions. Since there are still a large number of storage blocks waiting to be allocated to cache contents, we adopt multi-object auctions in our solution instead of single-object auctions to increase auction efficiency and reduce the number of auctions that needed in each allocation. We propose that, each of the RSUs provides one storage block to be auctioned in each multi-object auction. The detailed advantages of such method are discussed in Section IV-B.2.

B. Multi-Object Auctions Setup

In this subsection, we first propose a method to transform contents into equal-sized content blocks, then introduce our way to arrange a serial of multi-object auctions to complete the allocation, and finally provide the technique to calculate the evaluations of caching each content blocks into RSUs.

1) Transforming Contents Into Equal-Sized Content Blocks: To match the size of storage blocks that specified by the auctioneer, contents have to be adjusted to form equal-sized content blocks. Here we have to notice that, contents belonging to different CPs are not allowed to be put into one content block because CPs have to evaluate their own content blocks separately.

Since the formulated caching problem is a little bit similar to the classical knapsack problem [34], the transforming procedure we propose here is inspired by one of the greedy algorithm. For a given CP, we sort all of its contents in the descending order of *popularity to size ratio*, and put them

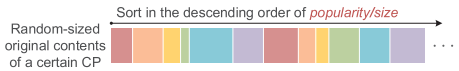


Fig. 4. The data ribbon of CP_l is formed by sorting its contents in the descending order of $\phi_{l,k}/S_{l,k}$.

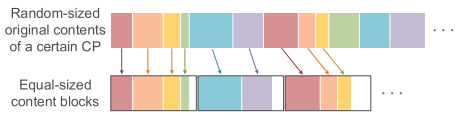


Fig. 5. A simple idea to transform contents into equal-sized content blocks without segmentation.

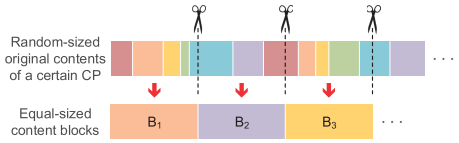


Fig. 6. The proposed method to transform contents into equal-sized content blocks with the consideration of segmentation.

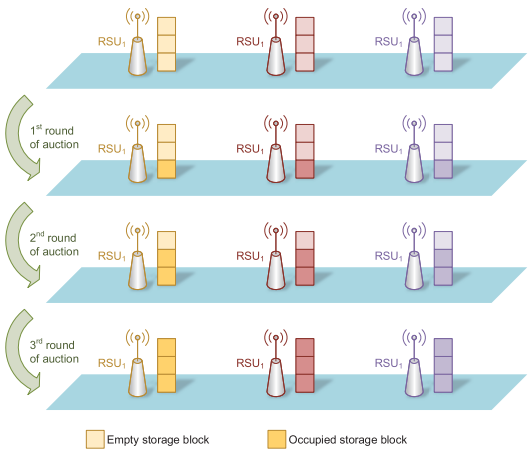


Fig. 7. The arrangement of a serial of multi-objet auctions, where the boxes represent the S_B-sized storages. The given example takes three rounds of auctions, which generate four states of the system, shown as the four rows.

together to form a one-dimensional long "data ribbon", as shown in Fig. 4.

However, there still could be different ways to transform this data ribbon into content blocks. A simple idea is to keep each content unsegmented, which is just to start from the left side, repeatedly put the current leftmost content into a S_B-sized block, and create a new empty block whenever the residual size of the current block is insufficient for the current content. Fig. 5 illustrates the procedure of such idea, where some of the space are being wasted. In contrast, the method we would like to propose is based on the assumption that contents can be segmented. And we only have to directly cut the data ribbon from the left side into multiple S_B-sized content blocks, as shown in Fig. 6. Here we ignore the minor problem that whether the length of the "data ribbon" can be divisible by S_B. This is because the most right side usually consists of low-popularity contents and they have little impact to the caching performance.

Due to the huge number of contents in reality, we recommend that S_B is set greater than the largest original content, in which way the computational complexity can be reduced to some extent. And as a result, each content is divided into no more than two content blocks. The larger S_B is set, the less number of contents are being segmented.

2) Arranging a Serial of Multi-Object Auctions: Recall that the storage capacity of RSU_m is S_m and the size of a storage block is S_B. Without the loss of generality, we assume that S_m can be divided by S_B. According to our scheme, there are totally $H = \max_m\{S_m/S_B\}$ rounds of multi-object auctions to finish the caching allocation. In the h^{th} round of auction, the h^{th} storage blocks in all the RSUs are being auctioned and allocated to cache content blocks. This process is essentially to auction the storages of all RSUs concurrently within multiple steps, as shown in Fig. 7.

The two main advantages of applying multi-object auctions instead of single object auctions are as follows: 1) Our scheme is more compatible with the case of multiple MNOs. If there are multiple MNOs who own different RSUs, then the auction sequence among different RSUs needs to be considered as long as single-object auctions are applied, because each single-object auction is only able to allocate a storage block of one RSU. Since different auction sequences can result in different caching results and different trading prices, fairness problem could arise among those MNOs. However, the proposed multi-object auction method is able to avoid such problems by allocating the storages of RSUs concurrently. 2) The communication costs can be reduced with multi-object auctions. The single-object auction needs $H \cdot M$ rounds of auctions, which causes $H \cdot M$ times of communications among CPs and MNOs. But the proposed method only needs H rounds of auctions, indicating a significant reduction. Either in multi-object auction or single-object auction, CPs have to submit the bids of its content blocks for all the RSUs in each auction, i.e., the costs of each time's communications are the same. Therefore, our scheme guarantees a much lower total communication costs to complete the caching allocation.

3) Approximate Valuations of Caching: In each multi-object auction, CPs have to calculate the marginal utility of caching its each content block into each RSU. Based on the estimations of system parameters, we are able to provide the way to calculate the marginal utility of caching each content block into each RSU. We first denote the b^{th} content block of P_l by $B_{l,b}$, which is a collection of content segments $\{C_{l,k,n}\}$. The marginal utility of caching $C_{l,k,n}$ into RSU_m in the t^{th} time slot is denoted by $V^{m,t}_{l,k,n}$. Based on current caching allocation, the amount of downloaded data of users downloading $C_{l,k}$ from two directions in the current time slot is $D_{1,l,k}$ and $D_{2,l,k}$, respectively. If we further cache $C_{l,k,n}$ into RSU_m, the result becomes $D^{t\prime}_{1,l,k}$ and $D^{t\prime}_{2,l,k}$, respectively. According to Eqn. (7), the marginal utility can be calculated as

$$V^{m,t}_{l,k,n} = \lambda\phi_{l,k}\left[f^t_1(D^{t\prime}_{1,l,k} - D^t_{1,l,k}) + f^t_2(D^{t\prime}_{2,l,k} - D^t_{2,l,k})\right].$$

Therefore, the final valuation of caching $B_{l,b}$ into RSU_m is

$$V^m_{l,b} = \sum_{t=1}^{T}\left[\sum_{C_{l,k,n}\in B_{l,b}} V^{m,t}_{l,k,n}\right]. \tag{15}$$

Before we go on to introduce the algorithm that uses valuations to generate the allocation results, we summarize the overall procedure of our solution in Algorithm 1.

Algorithm 1 The Overall Procedure of our Auction-Based Solution.

begin
  The auctioneer announces the block size S_B;
  Each CP transforms its own contents to form multiple S_B-sized content blocks;
  The total rounds of auctions is $H = \max_m\{S_m/S_B\}$;
  for *h is from* 1 *to* H **do**
    The h^{th} storage blocks in all RSUs are regarded as objects;
    Create the valuation matrix based on current allocation matrix;
    Run the market matching algorithm to complete a multi-object auction;
    Update the allocation matrix based on the result of the algorithm;
  end
end

C. Market Matching Algorithm

We adopt the *market matching algorithm* to determine the allocation results of each multi-object auction. This algorithm is originated from [36] and uses bipartite graphs to match the bidders and objects with maximum social welfare. That is to say, based on the results of previous rounds of auctions, the utility of the next-round's caching allocation can be maximized.

For convenience of reading, in this subsection, we use "content" and "storage" to represent "content block" and "storage block" respectively.

In general, this algorithm builds bipartite graphs based on the valuations and tries to find a perfect matching between contents and storages,[5] and finally outputs the auction results as the allocation scheme. Here, we describe this algorithm by a initialization procedure followed by an iteration procedure with four steps and finally a termination procedure, as given below:

1) Initialization Procedure: Given X contents and Y storages ($X > Y$), add $X - Y$ virtual storages. Then set the initial prices of all storages as zero, i.e., $p_y = 0$, $\forall y \in [1, X]$. The valuation of the x^{th} content for the y^{th} storage, $v_{x,y}$, is calculated according to Eqn. (15).

The introduction of virtual objects is to equalize the numbers of contents and storages, which is a prerequisite to find a perfect matching in a bipartite graph. Since virtual storages do not actually exist, the valuations for them are confined as zero in the algorithm. The content that matched to a virtual storage in the allocation results will not be allocated to any RSUs. Since the numbers of contents and storages are equalized, we can assume $X = Y$ in the following descriptions of this algorithm for clearer writing.

[5]In a matching, each content cannot be allocated to more than one RSU in each auction. And here we assume $M \leq H$ to avoid a possible unreasonable situation where a content with high popularity cannot be allocated into all of the RSUs. This assumption can be satisfied in most real-world situations because the storages can be divided into a large number of blocks.

Definition 1: A bipartite graph $G = (\mathcal{C}, \mathcal{S}, \mathcal{E})$ *is a graph that consists of two sets of nodes, $\mathcal{C}$, $\mathcal{S}$, and a set of edges $\mathcal{E}$, such that any edge in $\mathcal{E}$ connects one node from $\mathcal{C}$ with one node from $\mathcal{S}$.*

Definition 2: In $G = (\mathcal{C}, \mathcal{S}, \mathcal{E})$ where $\mathcal{C}$ stands for the content nodes and $\mathcal{S}$ stands for the storage nodes, given a fixed x, the edge between C_x and S_y exists if and only if $(v_{x,y} - p_y)$ is the largest for any y. Then such a bipartite graph is called a preferred-storage graph.

2) Iteration Procedure Step 1: Based on the valuations and the prices, a preferred-storage graph can be built. Intuitively, the preferred-storage graph shows the "most preferred" storages of each content. If the profit of acquiring storage S_y is the highest among all the storages for the content C_x, then there will be an edge between C_x and S_y. Note that, each single content may have multiple most preferred storages.

Definition 3: Given $G = (\mathcal{C}, \mathcal{S}, \mathcal{E})$, a matching $\mathcal{M}$ *is a subset edges of $\mathcal{E}$ such that each node in G is connected by no more than one edge in $\mathcal{M}$.*

Definition 4: Given $G = (\mathcal{C}, \mathcal{S}, \mathcal{E})$ and a matching $\mathcal{M}$ of it, an alternating path *consists of a serial of concatenated edges in $\mathcal{E}$ such that these edges are alternately included or excluded in $\mathcal{M}$.*

Definition 5: An augmenting path *is a special kind of alternating path where the two end-vertices are not connected by the edges in $\mathcal{M}$.*

3) Iteration Procedure Step 2: In a given preferred-storage graph, the algorithm tries to find augmenting paths and uses them to expand the matching until no more augmenting paths exist.

The classical breadth-first-search (BFS) algorithm in graph theory [37] can be applied to find augmenting paths by starting from any of the non-matched content nodes. We denote all the edges in the augmenting path as $\mathcal{E}'$ in a given bipartite graph with matching $\mathcal{M}$. Here $\mathcal{E}'$ consists of two disjoint subsets, $\mathcal{E}'_1$ and $\mathcal{E}'_2$, where $\mathcal{E}'_1$ is the set of edges included in $\mathcal{M}$ while $\mathcal{E}'_2$ is the opposite. By adding $\mathcal{E}_2$ to $\mathcal{M}$ and deleting $\mathcal{E}_1$ from $\mathcal{M}$, a greater matching can be formed, because $|\mathcal{E}_2| = |\mathcal{E}_1| + 1$. The matching achieves maximum as no more augmenting paths can be found.

4) Iteration Procedure Step 3: Based on the maximum matching in the last step, the algorithm decides whether to jump to the termination procedure. If all the nodes are matched, i.e., the maximum matching is a perfect matching, then jump to the **Termination Procedure** which ends the algorithm. Otherwise, the algorithm starts to find a constricted set, which is an obstruction for the existence of a perfect matching. The constricted set is defined as below:

Definition 6: In a preferred-storage graph, given $\mathcal{C}'$ as a subset of $\mathcal{C}$, all the directly connectable storage nodes from $\mathcal{C}'$ are denoted as the set $\mathcal{S}'$. If $|\mathcal{C}'| > |\mathcal{S}'|$, then $\mathcal{S}'$ is a constricted set.

Intuitively, a constricted set $\mathcal{S}'$ shows a situation where more contents are competing for less storages. Studies in [38] show that the equivalence condition of the existence of a perfect matching in a bipartite graph is the absence of constricted sets. The process of searching for a constricted set is based on the process of trying to find augmenting paths. When the

algorithm fails to find an augmenting path during BFS, the visited storage nodes form a constricted set [38].

5) Iteration Procedure Step 4: Once a constricted set $\mathcal{S}'$ is found, the algorithm raises the prices of storages in $\mathcal{S}'$ uniformly by a certain amount of value δp. Here δp is the minimum value to make at least one content change their preferred-storages so that a new preferred-storage graph can be built. After that, if the algorithm finds $\min\{p_y\} = \delta > 0$, then it lets $p_y = p_y - \delta$ for all y (which is necessary for the proof of the convergence property [36]). Thereafter, the algorithm goes back to **Iteration Procedure Step 1** for the next round of iteration.

6) Termination Procedure: The algorithm ends here. The matching in the bipartite graph shows the allocation of contents to storages, and the price vector represents the final trading prices. As a summary, we provide an overview of this algorithm in Algorithm 2.

Algorithm 2 Market Matching Algorithm to Solve Each Multi-Object Auction.

Input: Valuation matrix $V_{X\times Y}$ (X: number of contents, Y: number of storages, $X > Y$).
Output: Allocation matrix $\Delta_{X\times Y}$ and trading price vector P_Y.
Initialization Procedure:
Add $X - Y$ virtual objects and expand the valuation matrix from $V_{X\times Y}$ to $V_{X\times X}$ with 0;
Initialize the price vector P_X as zero;
Iteration Procedure:
  Build a preferred-storage graph $G(\mathcal{C}, \mathcal{S}, \mathcal{E})$ based on $V_{X\times X}$ and P_X;
  while *a augmenting path can be found during BFS* **do**
    Expand the current matching M according to the augmenting path;
  end
  if *M is a perfect matching* **then**
    Break the iteration;
  else
    Find a constricted set $\mathcal{S}'$ in $G(\mathcal{C}, \mathcal{S}, \mathcal{E})$ by BFS;
    Find the minimum additional price δp for storages in $\mathcal{S}'$ that can change G;
    Let $p_x = p_x + \delta p$ for all $S_x \in \mathcal{S}'$, and let $p_x = p_x - \min\{p_x\}$ for all $1 \le x \le X$;
  end
Termination Procedure:
Deduce matrix $\Delta_{X\times Y}$ from the matching, which shows the caching allocation;
The vector P_Y represents the final trading prices that have to be paid;

The convergence of this algorithm can be guaranteed as long as the valuations are quantified as decimals with finite precision and finite upper-bound [36]. The performance degradation brought by quantification is ignorable, as long as the value space has more than 100 quantized values (i.e., quantification accuracy $q \ge 100$) [39]. In additional, the practical complexity of running this algorithm can be far below the theoretical upper bound $O(N^4)$ where N is the number of content blocks.

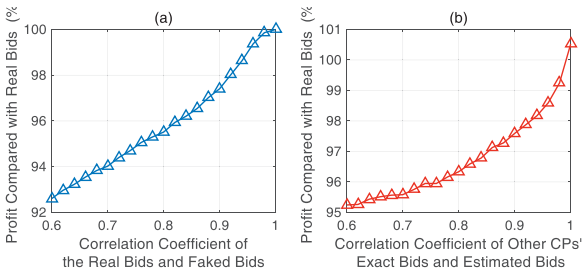


Fig. 8. (a) The average profit of faking bids with unknown others' bids. (b) The average profit of faking bids with estimated others' bids but also with errors. In both subplots $x = 1$ represents truthful bids.

D. Discussions on the Auction

In our paper, we use CP's true valuations as their bids in the auction. But whether truthful bidding remains to be their best bidding strategies is worth discussing. The original algorithm has a "truth-telling" property, where bidders' best strategy is to bid truthfully even with the knowledge of others' bids [40]. Its proof is based on the assumption that all the bidders are decoupled from each other. However, in our situation, we regard each content block as the corresponding bidder to be matched, and a CP who owns multiple content blocks is able to bid for all its content blocks jointly.

If we consider the auction as a "complete information game" where each CP is aware of all the other's exact valuations, then the outcome of truthful bidding is not a Nash equilibrium solution. However, the "complete information" assumption is not reasonable in reality. The proper assumption is to consider the auction as an "incomplete information game", where each CP at most know the statistical distribution of the others' bids. If we take the auctions as "incomplete information games", then we should consider "Bayesian Nash equilibrium" [41] as a substitute to analysis whether truthful bidding will be adopted by the CPs in the auction. Specifically, this means that from the point view of each certain CP, any one of the other CP has multiple possible "types" and will only choose one "type" to participate in the auction based on the probabilistic distribution of all its types. And for the situation in our paper, each CP has almost infinite number of types since the submitted bidding matrix has countless variations. If there is a Bayesian Nash equilibrium in our auction, it should be seen as the average outcome of infinite types.

Unfortunately, it is difficult to find the Bayesian Nash equilibrium theoretically even if we assume that all the CPs have the same probability distribution. Such difficulty is mainly due to the sensitivity of the market matching algorithm. Even the tiniest change of bidding matrix could bring significant difference to the outcome. As a result, the relation of "bidding-profit" of each CP is neither convex or concave—it is not even continuous.

However, the truthful bidding is likely to be such an equilibrium. This is because truth-telling averagely maximizes the expected profit of each CP, and any deviation from truthful bids brings averagely negative utility. Fig. 8 shows the outcome of 100 contents where half of the contents are possessed by the concerned CP. From the results, we can

TABLE I
SIMULATION PARAMETERS

Number of time slots in a day T	24
Vehicular density $u(t)$ in 24 hours respectively (m^{-1}) (based on data in [42])	0.190, 0.105, 0.055, 0.055, 0.070, 0.100, 0.150, 0.325, 0.500, 0.580, 0.650, 0.725, 0.705, 0.635, 0.605, 0.575, 0.400, 0.360, 0.420, 0.500, 0.550, 0.485, 0.305, 0.225
Maximum vehicular density ρ_{max} (m^{-1})	0.8
Maximum vehicular velocity u_{max} (ms^{-1})	20
Handoff delay t_h (ms)	from 50 to 1000
Length of the street L_s (m)	3500
Coverage length of RSUs l (m)	200
Number of RSUs M	from 20 to 30
Total content number K	from 10000 to 20000
Size of contents $S_{l,k}$ (GB)	uniformly distributed in $[0.1, 1]$
Size of blocks S_B (GB)	from 10 to 100
Storage capacity S (GB)	from 100 to 1000
Number of wireless channel U	7
Maximum rate of each channel R_{down} (MBytes/s)	2.5
Maximum backhaul rate R_{back} (MBytes/s)	2.5
Proportion of interested users λ	from 0.1 to 0.5
Traffic coefficient α	1
Zipf coefficient β	1
Quantification accuracy q	1000

conclude that fake bidding strategies averagely bring lower profit. In addition, only a small amount of estimation error of others' bids also causes significant decrease in profit. From a practical point of view, the attempts to predict others' bids and calculate better strategies could also be troublesome for the CPs, which should be considered as a non-negligible cost. Therefore, the CPs are not likely to lie about the true valuations in the auction, and the allocation results based on truthful bids can be considered as the outcome of the results of an auction in real-world.

V. SIMULATION RESULTS

In this section, we simulate the performance of the proposed auction-based solution and verify the impacts of system parameters that discussed in Section III. The simulation parameters are set in the first subsection. The simulation results and corresponding discussions are provided in the second subsection.

A. Simulation Parameters

Without loss of generality, we set $S_m = S$ for all $1 \leq m \leq M$ and $\rho_1^t = \rho_2^t$ for all $1 \leq t \leq T$. In addition, we choose one hour to be the length of a time slot, which makes $T = 24$. Although a shorter time slot would make the results more accurate, our key point is only to solve the problem with a given length of time slot. According to [42], the traffic loads in different days have the similar profile, so we set the variation of vehicular density ρ^t in a similar way, as given in the second row in Table I. The popularity of contents conform to Zipf-like distribution, which is to say, the popularity of the k^{th} most popular content is $\phi(k) = \frac{Z}{k^\beta}$, where $Z = \sum_{k=1}^{K} \frac{1}{k^\beta}$ is the normalization factor and β is the Zipf coefficient, which is usually around 1.

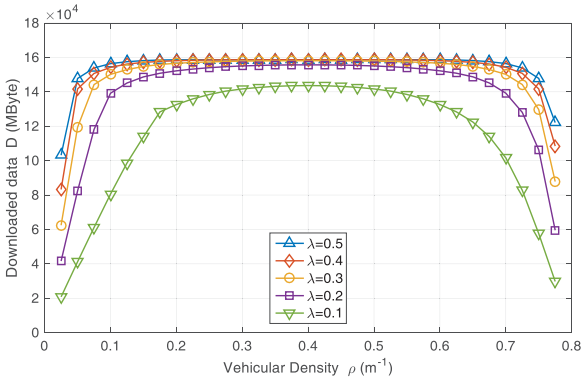


Fig. 9. The impacts of the vehicular density ρ and the proportion of interested users λ, where $t_h = 50ms$, $S = 1000GB$, $S_B = 20GB$, $M = 25$ and $K = 10000$. The Y-axis shows the amount of data downloaded in each hour with given ρ and λ.

Table I shows a more detailed list of parameters in our simulations.

B. Simulation Results and Discussions

In this subsection, we first simulate the vehicular density ρ to see its impact on each hour's performance. A 24-hour situation is then provided to show the performance's variation in a day. After that, the influence of the size of capacity S, the number of contents K, the size of blocks S_B, the number of RSUs M and the handoff delay t_h are given. In some of the simulations, we also compare our solution to several related solutions to show the advantages of ours.

As the vehicular density is significant to the system performance in each hour, our first simulation presents the impact of ρ. Fig. 9 shows the corresponding results, where handoff delay $t_h = 50ms$, RSUs' capacity $S = 1000GB$, size of block $S_B = 20GB$, number of RSUs $M = 25$ and number of contents $K = 10000$. The Y-axis represents the amount of downloaded data D in each hour according to the given ρ on the X-axis. When the value of ρ gets closer to 0 or ρ_{max}, the amount of downloaded data per hour drastically decreases. As ρ is somewhere in the middle of $(0, \rho_{max})$, the curve of D keeps flat and does not response much to the variation of ρ. Such behavior can be explained by the proof of Proposition 3, where the concave feature is predicted. Intuitively speaking, for the case where ρ is close to 0, the lack of users results in the lack of data being downloaded. In contrast, for the case where ρ is close to ρ_{max}, the crowded vehicles reduce the total flux of vehicles and thereby also reduce the total data being downloaded. This phenomenon implies that the performance of the RSU caching system can be fully exploited when the density of vehicles is moderate.

Fig. 9 also gives the results with different settings of the proportion of interested users λ. We can observe that with more users participating into content downloading, the amount of total downloaded data increases, which also accords with Proposition 7. However, it is noticeable that the total amount of downloaded data will not unlimitedly increase by enlarging λ. The system seems to have an upper bound, implying the maximum ability of providing data services. The actual

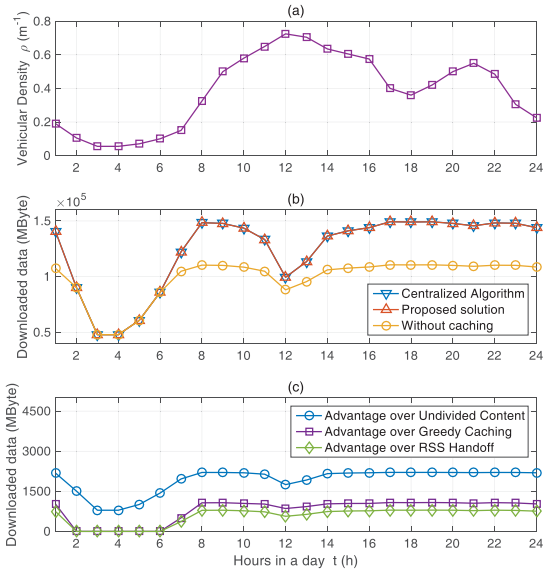


Fig. 10. Subplot (b) and (c) show the variation of the amount of downloaded data in each hour with the given setting of ρ in subplot (a), with $\lambda = 0.1$, $t_h = 50ms$, $S = 1000GB$, $S_B = 20GB$, $M = 25$ and $K = 10000$.

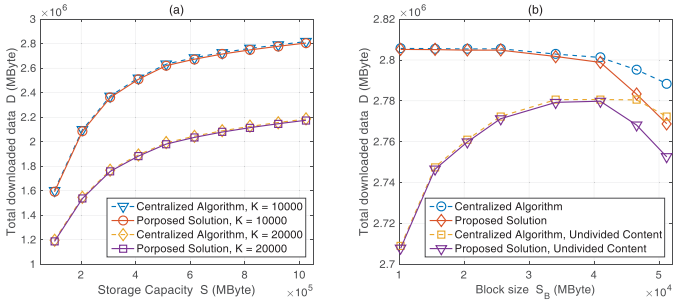


Fig. 11. (a) The impacts of number of contents K and the capacity of RSUs S, where $\lambda = 0.1$, $t_h = 50ms$, $S_B = 20GB$ and $M = 25$. (b) The impact of the size of blocks S_B, where $\lambda = 0.1$, $t_h = 50ms$, $S = 1000GB$, $K = 10000$ and $M = 25$.

downloaded data is the closest to such an upper bound while the vehicular density $\rho_{up} = \rho_{max}/2$. Note that this feature is originated from the adopted vehicular model, just as shown in Fig. 2 (b). A different choice of α will change the value of ρ_{up}.

Based on the analysis of the impact of vehicular density, we can simulate a one-day situation where 24 time slots are included, as shown in Fig. 10, where $\lambda = 0.1$, $t_h = 50ms$, $S = 1000GB$, $S_B = 20GB$, $M = 25$ and $K = 10000$. Fig. 10 (a) shows the setting of the vehicular density in different hours, which mainly has two peaks: at noon and in the evening. The maximum vehicular density is around 0.75 at twelve o'clock, which means the traffic is crowded by then. Fig. 10 (b) illustrates the system performance of caching during a whole day. Based on the results of Fig. 9, we are able to explain the two valleys in the curve. The former one results from the sparsity of vehicles and the latter one is due to the congestion of the vehicular traffic. It is also noticeable that the small peak at 21 o'clock in Fig. 10 (a) is not reflected in Fig. 10(b), since the vehicular density is not high enough to generate obvious degradation of the performance. As a result, the performance in most hours stays still until vehicular density goes over or below some thresholds. In the given setting, these thresholds are approximately $0.2m^{-1}$ and $0.7m^{-1}$. Fig. 10 (b) also reveals the performance gains of our auction-based solution and the centralized algorithm mention in Section III-B. This centralized algorithm takes $M \times H$ rounds of allocation, where in each round the greatest value in the valuation matrix is selected and corresponding content is added in the corresponding RSU. It was proved to be at least $(1 - 1/e)$ of the performance of a optimal one [33], and we use it to be a benchmark in the simulation. The advantage of the centralized algorithm is almost unnoticeable and they both can elevate the system performance in most time of the day. In the midnight when there is little traffic in the street, the advantages brought by caching disappear. That is to say, the RSU caching system is unnecessary if a street is nearly empty, and the cache-enabled RSUs should be deployed in the street that with appropriate traffic.

Fig. 10 (c) illustrates the advantage of our solution over some related ones. The *Undivided Content* is based on the simple idea to transform contents into content blocks, as shown in Fig. 5 in Section IV-B.1, which keeps contents unsegmented. The *Greedy Caching* is another centralized solution, which caches the contents according to the descending order of "popularity to size ratio", and in fact does not transform the contents into content blocks. The *RSS handoff* uses the same auction solution with our proposed one, except for the condition that the handoff decision is based on a traditional one, as mentioned in Section III-A. The overall advantages of our solution to those three are positive. The advantage over *Undivided Content* comes from the full utilization of caching space because *Undivided Content* wastes some of the caching storages by keeping contents unsegmented. The advantage over *Greedy Caching* shows that our auction-based solution is better than a simple and straight-forward centralized caching scheme. The advantage over *RSS handoff* is brought by the adopted handoff decision mechanism, which improves the utilization of overlapping areas of RSUs.

The impacts of number of contents K and the capacity of RSUs S are shown in Fig. 11 (a), where $\lambda = 0.1$, $t_h = 50ms$, $S_B = 20GB$ and $M = 25$. Note that the Y-axis is the total downloaded data in 24 hours instead of in each single hour. As Propositions 4 and 5 predict, D has a positive correlation with S and a negative correlation with K, which agrees with our simulation. It can be observed that the curve of $K = 10000$ is high above the curve of $K = 20000$, and the two curves have a similar shape. Thus a greater number of contents K makes it more difficult to achieve high performance. We can also see that when B gets greater, D increases but the change rate of D decreases as well. Therefore, the same amount of storage makes greater difference in a low-capacity situation.

The impact of the size of blocks S_B is shown in Fig. 11 (b), where $\lambda = 0.1$, $t_h = 50ms$, $S = 1000GB$, $K = 10000$ and $M = 25$. The two upper curves show the results of contents that can be segmented, while the two lower curve shows the outcome of the *Undivided Content*. For the proposed solution, the larger S_B is, the worse the outcome becomes. This is because a larger content block combines too many contents together and these contents have to be allocated as a whole, which reduces the flexibility of allocation and

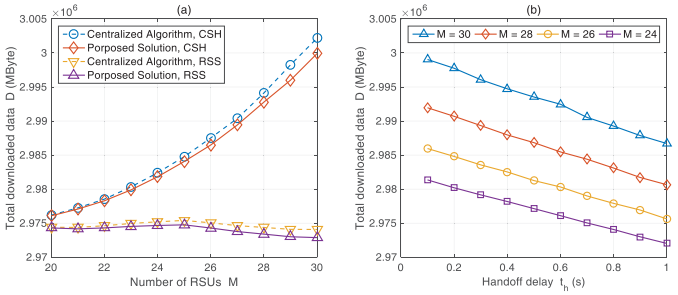


Fig. 12. (a) The impact of the number of RSUs M, where $\lambda = 0.1$, $t_h = 50ms$, $S = 1000GB$, $S_B = 20GB$ and $K = 10000$. (b) The impact of the handoff delay t_h, where $\lambda = 0.1$, $S = 1000GB$, $S_B = 20GB$ and $K = 10000$.

deviates more from the optimal solution (a hypothetical one). A smaller S_B can improve the caching efficiency but also induce higher computational complexity. Thus a practical choice of S_B should take both factors into account. The outcome of *Undivided Content*, however, is not monotonous of S_B. The reason is that the average wasted space brought by unsegmented contents decreases while the size of blocks gets larger. Thus, the difference between the two curves gets fewer when we increase S_B. It can be also observed that the difference of our solution and the centralized solution is not noticeable until S_B is larger than $4 \times 10^4 MBytes$. The reason is provided in a footnote in Section IV-C, where we need to guarantee $S/S_B \geq M$ to avoid an unreasonable situation. Therefore, to guarantee the performance of the auction-based solution, it is necessary to choose a smaller S_B.

In Fig. 12 (a), we illustrate the impact of the number of RSUs M, where $\lambda = 0.1$, $t_h = 50ms$, $S = 1000GB$, $S_B = 20GB$ and $K = 10000$. The outcome of the traditional handoff is not sensitive to the number of RSUs as the two lower curves in this figure is almost flat. For the adopted handoff decision where users choose the cached RSUs as priority, the total amount of data increases as more RSUs are being deployed. In addition, the change rate of D gets larger while M continues to increase. However, the improvement is not significant, where 10 more RSUs only lead to 1% performance increased. This is mainly because adjacent RSUs can also cause interference with each other (Remember that in Section II we assume that the user in the overlapping area takes up both RSUs' channel due to interference). Therefore, we can deduce that it is not an effective way to improve the performance by adding additional RSUs in the street. Since the difference of the two handoff methods is not significant, we can conclude that even a traditional handoff will not degrade much of the system performance.

Finally, we provide the impact of the handoff delay t_h in Fig. 12 (b), where $\lambda = 0.1$, $S = 1000GB$, $S_B = 20GB$ and $K = 10000$. As can be observed in this figure, for each setting of M (given by 30, 28, 26 and 24, respectively), D has an approximate linear correlation with t_h. The reason is that t_h influences the performance by deferring the procedure of switching an uncached RSU to a cached RSU. Such a loss can be approximately calculated as $L = t_h(r_{cache} - r_{uncache})$. The uneven density of the four curves in this figure results from the non-linear property of the upper curve in Fig. 12(a). Based on some of the existing studies like [30], the handoff delay between RSUs is able to be reduced to about $10ms$. Therefore, our caching system is able to keep its efficiency in a practical situation with the consideration of handoff.

VI. Conclusion

In this paper, we studied a roadside caching scenario where multiple CPs intended to cache their own contents into the storages of RSUs for better data dissemination. We focused on a single two-way street and formulated the caching problem as how to maximize the total downloaded data by vehicular users. In the theoretical analysis, we first predicted the downloaded data as a concave function of vehicular density. Then we found the number of RSUs M and the proportion of interested users λ had positive impacts on the performance, while the number of contents K, the size of blocks S_B and the handoff delay t_h had negative impacts. To solve the caching problem with the consideration of the competition of CPs, we designed an auction-based solution, which included a serial of multi-object auctions to complete the allocation. Simulation results testified the theoretical analysis and also revealed a satisfying performance of our solution. The advantages of caching-specific handoff and segmented contents was also verified.

References

[1] Z. Hu, Z. Zheng, T. Wang, and L. Song, "Roadside Unit Caching Mechanism for Multi-Service Providers," in *Proc. ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Hangzhou, China, Jun. 2015, pp. 387–388.

[2] G. Karagiannis *et al.*, "Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions," *IEEE Commun. Surveys Tuts.*, vol. 13, no. 4, pp. 584–616, 4th Quart., 2011.

[3] H. T. Cheng, H. Shan, and W. Zhuang, "Infotainment and road safety service support in vehicular networking: From a communication perspective," *Mech. Syst. Signal Process.*, vol. 25, no. 6, pp. 2020–2038, Aug. 2011.

[4] M. Amadeo, C. Campolo, and A. Molinaro, "Enhancing content-centric networking for vehicular environments," *Comput. Netw.*, vol. 57, no. 16, pp. 3222–3234, Nov. 2013.

[5] M. Chaqfeh, A. Lakas, and I. Jawhar, "A survey on data dissemination in vehicular ad hoc networks," *Veh. Commun.*, vol. 1, no. 4, pp. 214–225, Oct. 2014.

[6] S. Abdelhamid, H. S. Hassanein, G. Takahara, and H. Farahat, "Caching-assisted access for vehicular resources," in *Proc. Annu. IEEE Conf. Local Comput. Netw.*, Edmonton, AB, Canada, Sep. 2014, pp. 28–36.

[7] S. Abdelhamid, H. S. Hassanein, and G. Takahara, "On-road caching assistance for ubiquitous vehicle-based information services," *IEEE Trans. Veh. Technol.*, vol. 64, no. 12, pp. 5477–5492, Sep. 2015.

[8] N. Kumar and J.-H. Lee, "Peer-to-peer cooperative caching for data dissemination in urban vehicular communications," *IEEE Syst. J.*, vol. 8, no. 4, pp. 1136–1144, Dec. 2014.

[9] T. H. Luan, L. X. Cai, J. Chen, X. S. Shen, and F. Bai, "Engineering a distributed infrastructure for large-scale cost-effective content dissemination over urban vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 3, pp. 1419–1435, Mar. 2014.

[10] R. Ding, T. Wang, L. Song, Z. Han, and J. Wu, "Roadside-unit caching in vehicular ad hoc networks for efficient popular content delivery," in *Proc. IEEE Wireless Commun. Netw. Conf.*, New Oleans, LA, USA, Mar. 2015, pp. 1207–1212.

[11] L. Idir, S. Paris, and F. Naït-Abdesselam, "Optimal caching of encoded data for content distribution in vehicular networks," in *Proc. IEEE Int. Conf. Commun. Workshop*, London, U.K., Jun. 2015, pp. 2483–2488.

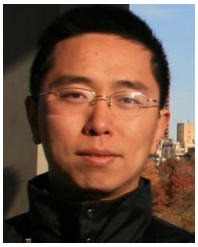
[12] B. B. Chen and M. C. Chan, "MobTorrent: A framework for mobile Internet access from vehicles," in *Proc. IEEE Int. Conf. Comput. Commun.*, Rio de Janeiro, Brazil, Apr. 2009, pp. 1404–1412.

[13] U. Shevade, Y.-C. Chen, L. Qiu, Y. Zhang, V. Chandar, M. K. Han, H. H, Song, and Y. Seung, "Enabling high-bandwidth vehicular content distribution," in *Proc. ACM Int. Conf.*, Philadelphia, PA, USA, Nov. 2010, pp. 1–12.

[14] K. Mershad and H. Artail, "SCORE: Data Scheduling at roadside units in vehicle ad hoc networks," in *Proc. Int. Conf. Telecommun.*, Jounieh, Lebanon, Apr. 2012, pp. 1–6.
[15] F. Malandrino, C. Casetti, C.-F. Chiasserini, and M. Fiore, "Optimal content downloading in vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 12, no. 7, pp. 1377–1391, Jul. 2013.
[16] P. Salvo, F. Cuomo, A. Baiocchi, and A. Bragagnini, "Road side unit coverage extension for data dissemination in VANETs," in *Proc. Annu. Conf. Wireless On-Demand Netw. Syst. Services*, Courmayeur, Italy, Jan. 2012, pp. 47–50.
[17] Z. Dong, S. H. Dau, C. Yuen, Y. Gu, and X. Wang, "Delay minimization for relay-based cooperative data exchange with network coding," *IEEE/ACM Trans. Netw.*, vol. 23, no. 6, pp. 1890–1902, Dec. 2015.
[18] X. Wang, C. Yuen, T. J. Li, W. Song, and Y. Xu, "Minimizing transmission cost for third-party information exchange with network coding," *IEEE Trans. Mobile Comput.*, vol. 14, no. 6, pp. 1218–1230, Jun. 2015.
[19] X. Wang, C. Yuen, and Y. Xu, "Coding-based data broadcasting for time-critical applications with rate adaptation," *IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2429–2442, Jun. 2014.
[20] Z. Hu, Z. Zheng, T. Wang, L. Song, and X. Li, "Game theoretic approaches for wireless proactive caching," *IEEE Commun. Mag.*, vol. 54, no. 8, pp. 37–43, Aug. 2016.
[21] B. Yu and F. Bai, "ETP: Encounter transfer protocol for opportunistic vehicle communication," in *Proc. IEEE Int. Conf. Comput. Commun.*, Shanghai, China, Apr. 2011, pp. 2201–2209.
[22] V. Krishna, *Auction Theory*. San Diego, CA, USA: Academic, 2009.
[23] C. Yi and J. Cai, "Two-stage spectrum sharing with combinatorial auction and Stackelberg game in recall-based cognitive radio networks," *IEEE Trans. Commun.*, vol. 62, no. 11, pp. 3740–3752, Nov. 2014.
[24] C. Yi and J. Cai, "Multi-item spectrum auction for recall-based cognitive radio networks with multiple heterogeneous secondary users," *IEEE Trans. Veh. Technol.*, vol. 64, no. 2, pp. 781–792, Feb. 2015.
[25] C. Liang and F. R. Yu, "Wireless network virtualization: A survey, some research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 1, pp. 358–380, 1st Quart., 2014.
[26] P. Blasco and D. Gündüz, "Learning-based optimization of cache content in a small cell base station," in *Proc. IEEE Int. Conf. Commun.*, Sydney, NSW, Australia, Jun. 2014, pp. 1897–1903.
[27] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *Proc. IEEE Int. Conf. Comput. Commun.*, New York, NY, USA, Mar. 1999, pp. 126–134.
[28] B. Piccoli and A. Tosin, "Vehicular Traffic: A Review of Continuum Mathematical Models," in *Mathematics of Complexity and Dynamical Systems*. New York, NY, USA: Springer, 2009.
[29] B. Ma and X. Liao, "Speed-adaptive vertical handoff algorithm based on fuzzy logic in vehicular heterogeneous networks," in *Proc. Int. Conf. Fuzzy Syst. Knowl. Discovery*, Sichuan, China, May 2012, pp. 371–375.
[30] T.-Y. Wu, W.-T. Lee, F.-H. Liu, H.-L. Chan, and T.-H. Lin, "An efficient pre-scanning scheme for handoff in Cooperative Vehicular Networks," in *Proc. IEEE Int. Symp. Pers., Indoor Mobile Radio Commun.*, Toronto, ON, Canada, Sep. 2011, pp. 583–587.
[31] D. Kwak, J. Mo, and M. Kang, "Investigation of handoffs for IEEE 802.11 networks in vehicular environment," in *Proc. Int. Conf. Ubiquitous Future Netw.*, Hong Kong, Jun. 2009, pp. 89–94.
[32] T H. Corman, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, U.K.: MIT Press, 2001.
[33] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*. Berlin, Germany: Springer-Verlag, 2003.
[34] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*. New York, NY, USA: Wiley, 1990.
[35] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[36] G. Demange, D. Gale, and M. Sotomayor, "Multi-item auctions," *J. Political Econ.*, vol. 94, no. 4, pp. 863–872, Aug. 1986.
[37] D. B. West, *Introduction to Graph Theory*. Upper Saddle River, NJ, USA: Prentice-Hall, 2001.
[38] L. Lovász and M. D. Plummer, *Matching Theory*. Providence, RI, USA: AMS, 2009.
[39] Z. Hu, Z. Zheng, T. Wang, and L. Song, "Caching as a service: Small-cell caching mechanism design for service providers," *IEEE Trans. Wireless Commun.*, vol. 15, no. 10, pp. 6992–7004, Oct. 2016.
[40] H. B. Leonard, "Elicitation of honest preferences for the assignment of individuals to positions," *J. Political Econ.*, vol. 91, no. 3, pp. 461–479, Jun. 1983.
[41] D. Fudenberg and J. Tirole, *Game Theory*. Cambridge, U.K.: MIT Press, 1991.
[42] J. Lorincz, T. Garma, and G. Petrovic, "Measurements and modelling of base station power consumption under real traffic loads," *Sensors*, vol. 12, no. 4, pp. 4281–4310, Mar. 2012.

Zhiwen Hu (S'15) received the B.S. degree in electronic engineering from Peking University, China, in 2015, where he is currently pursuing the Ph.D. degree with the School of Electrical Engineering and Computer Science. His current research interests include caching in wireless networks, game theory in 5G networks, and wireless M2M communications.

Zijie Zheng (S'14) received the B.S. degree in electronic engineering from Peking University, China, in 2014, where he is currently pursuing the Ph.D. degree with the School of Electrical Engineering and Computer Science. His current research interests include game theory in 5G networks, wireless powered networks, and mobile social networks.



Tao Wang (SM'11) received the B.S. and Ph.D. degrees from Peking University in 1999 and 2006, respectively. He was a Post-Doctoral Researcher with Intel from 2006 to 2008, where he was a Staff Research Scientist. He is currently an Associate Professor with the School of Electronics Engineering and Computer Science, Peking University. He is also a CCF (Chinese Computer Society) Senior Member. He joined Peking University as a Faculty Member in 2010. In the past five years, he received many research fundings and published about 30 research papers in top journals/conferences. His current research interests are computer architecture, reconfigurable wireless network architecture, and wireless e-health. He received the Intel China Employee of the Year Award in 2008, which is the highest individual award at Intel China.

Lingyang Song (S'03–M'06–SM'12) received the Ph.D. degree from the University of York, U.K., in 2007. He was a Research Fellow with the University of Oslo, Norway, until rejoining Philips Research, U.K., in 2008. In 2009, he joined the School of Electronics Engineering and Computer Science, Peking University, China, as a Full Professor. His main research interests include MIMO, cognitive and cooperative communications, security, and big data. He was a recipient of the IEEE Leonard G. Abraham Prize in 2016 and IEEE Asia Pacific Young Researcher Award in 2012. He is currently on the Editorial Board of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS. He received the K. M. Stott Prize for excellent research from the University of York.

Xiaoming Li (S'87–M'87–SM'03) received the Ph.D. degree in computer science from the Stevens Institute of Technology, NJ, USA. He is currently a Professor with Peking University, China. His research interests include search engine and web mining. He is a fellow of the Computer Federation of China (CCF) and a member of Eta Kappa Nu. He serves on the Editorial Board of *Concurrency and Computation* (John Wiley). He received the CCF Wang Xuan Award and the Outstanding Educator Award in 2013 and 2014, respectively.