Low Cost Video Streaming through Mobile Edge Caching: Modelling and Optimization

Luigi Vigneri, *Student Member, IEEE,* Thrasyvoulos Spyropoulos, *Member, IEEE,*
and Chadi Barakat, *Senior Member, IEEE*

***Abstract*—Caching content at the edge of mobile networks is considered as a promising way to deal with the data tsunami. In addition to caching at fixed base stations or user devices, it has been recently proposed that an architecture with public or private transportation acting as mobile relays and caches might be a promising middle ground. While such mobile caches have mostly been considered in the context of delay tolerant networks, in this paper we argue that they could be used for low cost video streaming *without the need to impose any delay on the user*. Users can prefetch video chunks into their playout buffer from encountered vehicle caches (at low cost) or stream from the cellular infrastructure (at higher cost) when their playout buffer empties while watching the content. Our main contributions are: (i) to model the playout buffer in the user device and analyze its idle periods which correspond to bytes downloaded from the infrastructure; (ii) to optimize the content allocation to mobile caches, to minimize the expected number of non-offloaded bytes. We perform trace-based simulations to support our findings showing that up to 60 percent of the original traffic could be offloaded from the main infrastructure.**

I. Introduction

The last decade has seen an exponential growth in the mobile traffic demand [2]. To keep up with this demand, increasing the number of base stations (BSs) and caching popular content at such BSs have widely been considered. *Densification* improves spectral efficiency, while *edge caching* ensures that many requests can be served locally, reducing latency and offloading the backhaul/core network [32]. Nevertheless, installing many small cells (SCs) still requires high capital and operational expenditures, making operators reluctant due to stagnating revenues per user. Using user equipments (UEs) for storage and relaying (e.g., through device-to-device communication) largely avoids these costs, but is stifled by device resource constraints (especially battery) and privacy concerns. More recently, the use of private or public transportation means (e.g., cars, buses, taxis) (in addition to fixed SCs) as mobile SCs and storage points has been considered [1], [34]. We refer to such a storage cloud as *vehicular cloud*. Using vehicles as low cost relays and caches offers an interesting middle ground between fixed SCs and UEs as: (i) they are widespread; (ii) they can be equipped with storage, communication, computational capabilities at a lower cost than SCs; (iii) they have few resource limitations.

Nevertheless, short range nodes (i.e., SCs, UEs or vehicles) do not provide full coverage. A request will be offloaded only if (i) the UE is within range of a short range node *and* (ii) that node stores the requested content. To offload more traffic, researchers have suggested to introduce *delay tolerance*: a request that cannot be served immediately by a low(er) cost node will wait up to a deadline until such a low cost node (with the content) is encountered [34]. Unfortunately, the required access delays (up to hours) quoted in most related studies [5], [34] is not acceptable except for specific applications (e.g., software updates) or for low cost data plans.

In this paper, we propose to exploit the fact that *streaming of stored video content offers some delay tolerance "for free"*. Video content is split into many chunks that are streamed into the user's playout buffer one by one while consumed in parallel at the playout speed. The user does not have to wait until the whole content is found in a local cache, but can start streaming right away fetching chunks from the infrastructure or local/mobile caches depending on the availability of the latter and on the buffer status. For example, if a user is watching a one-hour video, the chunks corresponding to the x^{th} minute of the video do not have to be downloaded until right before that time. During that time, a mobile cache with that chunk might be encountered and these bytes can be offloaded *without any impact to user experience*[1]. In this context, two interesting questions arise:

1) How many bytes of streamed video get offloaded through such mobile edge caching?
2) How can we optimize the edge cache allocation to maximize the amount of bytes that get offloaded?

The goal of this paper is to provide initial answers to these questions assuming vehicular nodes acting as mobile SCs and local caches, and streamed video-on-demand content as the main application[2]. Our main contributions are the following:

- We model the playout buffer dynamics of a user device as a queueing system, and analyze the expected amount of offloaded traffic (corresponding to the idle periods of this buffer) as a function of network characteristics (e.g., vehicle density, file characteristics) and a given cache allocation.
- Based on this model, we formulate the problem of optimal content allocation in vehicles to minimize the total load on the cellular infrastructure. Then, we show that the problem is NP-hard, and propose appropriate approximations for two interesting regimes of vehicular traffic densities.
- Next, based on this result, we propose a two-phase per-chunk allocation policy, where the available space is allo-

[1]Note that while chunks are also "downloaded" in the streaming case, the difference here is that no extra delay needs to be *imposed* on the user (and thus deteriorate her quality of experience). Any chunk not available in the playout buffer when its playout time arrives can be retrieved from main infrastructure.

[2]Note that this scenario does not include live content streaming, which is not usually amenable to caching, and is often optimized using multicast.

cated first among different content (according to the previous policy), then the space for a specific content is allocated among its chunks, taking into account chunk popularity.
- We validate our theoretical results using real traces for content popularity and vehicle mobility, and show that our system can offload up to 60 percent of streamed data in realistic scenarios, even with modest technology penetration.

As a final remark, while we present our analysis within the context of vehicular relays, the main framework and a number of results could be applied to content streaming from fixed SCs or even UEs (we discuss such generalizations in Section VII).

The rest of the paper is organized as follows: in Section II, we review the existing literature about mobile edge caching focusing on video streaming. In Section III, we introduce the system assumptions. In Section IV, we formulate and solve the optimization problem at hand for different vehicular traffic regimes. In Section V, we introduce the per-chunk allocation policy. Then, we perform real trace-based simulations in Section VI. Finally, we discuss about additional applications in Section VII, and we conclude our paper in Section VIII with a summary and future work.

II. Related work

The exponential growth of the mobile traffic is pushing academic research and companies to study new solutions to decrease the load on the cellular network: specifically, on the one hand, the networking research community is more interested in the theoretical problems of, e.g., content allocation, modelling, mobility pattern analysis; on the other hand, mobile network operators (MNOs) see such a problem as an opportunity to fill the market gap with new products. In this section, we present a list of the most relevant work in edge caching.

A. Caching at SCs

SCs constitute a promising solution to deal with the mobile data growth that is overloading the cellular network. Local caching of popular content items at the SC BSs has been proposed to decrease the costly transmissions from the macrocell BSs without requiring high capacity backhaul links for connecting the SCs with the core network. In this context, traditional solutions concern adding storage capacity to SC BSs and/or to WiFi access points with a potential introduction of delay tolerance [5], [21], [40] to further increase the number of cache hits. The femtocaching idea is proposed as a solution to compensate for the weak backhaul capacity by deploying coverage-limited nodes with high storage capacity called femtocaches. The basic idea of caching in SCs has first been presented in Golrezaei *et al.* [11]. Their main challenge is to analyse the optimum way of assigning content to the SC BSs (called *helpers*) to minimize the expected download time or maximize the number of cache hits. Such a generic formulation has later been improved and extended by several researchers considering different assumptions and models. For instance, Zhang *et al.* [36] propose to cache content in wireless access points based on popularity. Also, the authors propose to separately add a prefetch buffer to capture short-term content access patterns using aggregated network-level statistics.

While such distributed caching schemes for SCs provide very interesting theoretical insights and algorithms, they face some key shortcomings. A large number of SCs is required for an extensive enough coverage by SCs, which comes at a high cost [4]. Conversely, vehicles are widespread, mainly in urban environments. Furthermore, the smaller size of edge caches and smaller number of users per cell raises the question whether enough overlap in user demand would be generated locally to have a high enough hit ratio, when real traffic is considered. Delayed content access is supposed to overcome such a limitation [5], [21]. On the other hand, caching video streaming does not require additional delays that would degrade the user quality of experience (QoE).

B. Video caching

A number of works have recently focused their interest on caching of video content since multimedia files have become popular. For instance, Poularakis *et al.* [25] optimize the service cost and the delivery delay. In their work, prestored video files can be encoded with two different schemes in various qualities. Differently, Dandapat *et al.* [9] study just in time video streaming. The authors propose to exploit WiFi access points in cities to build a large distributed video caching system for smooth streaming. Since a user is not able to download an entire video from the same hotspot, the authors promote replication of video chunks across access points while aiming at minimizing this replication (efficient content storage). Furthermore, Ahlehagh and Dey [3] improve video capacity and user experience of mobile networks. The authors propose video-aware backhaul and wireless channel scheduling techniques to maximize the number of concurrent video sessions by satisfying QoE requirements. They consider video QoE as consisting of initial buffering delay and number of stalls during the video session.

In this paper, we introduce a model to deal specifically with multimedia content. This model is based on queueing theory and handles a number of system parameters such as heterogeneous content size, vehicle mobility and limited content capacity. According to such a model that exploits the intrinsic delay tolerance of later chunks, we are able to formulate an optimization problem to allocate video content to caches, and solve it efficiently. The main novelties are: (i) the framework has a wide applicability since its generic formulation can be easily adapted to similar offloading scenarios such as femtocaching or caching on mobile devices; (ii) we combine the ideas of video caching at the edge and vehicular networks to improve the percentage of traffic offloaded; (iii) we provide insights on the optimal allocation for the optimization problem for different types of vehicle densities.

C. Caching at vehicles

Recent technology advances allow car manufacturers to build vehicles smarter and more sophisticated. New vehicles are able to communicate with each other to exchange information about security and traffic, and provide infotainment systems to passengers. While large setup delays might be an obstacle, recent protocols (e.g., DSRC) have been considerably

boosting vehicular networks over the last ten years. For this reason, MNOs see vehicles as (i) new potential customers to connect to their network, and dedicated data plans have been proposed. What is more, WiFi offloading for moving vehicles poses unique challenges due to high mobility, and researchers have been interested to model and analyze such an environment [7]. For instance, Mahmood *et al.* [22] introduce a probabilistic model to cache content at the edge nodes in order to serve content requests from vehicles (or moving users). As an alternative, vehicles can be used as (ii) nodes to boost the current cellular infrastructure. First works in this direction have appeared in the late 2000s [37], [38]. Zhang *et al.* [37] propose a peer-to-peer scheme to improve the performance of content sharing in intermittently connected vehicular networks. Zhao and Cao [38] adopt the idea of carry and forward content where a moving vehicle carries information until a new vehicle moves into its vicinity: the authors make use of the predictable vehicle mobility to reduce the content delivery delay. A more recent work [29] proposes erasure coding to reduce latency in vehicular networks which, however, requires large caches to be efficient. The hype around vehicular networks as part of the cellular infrastructure has been confirmed by the interest of car manufacturers and by the launch of new companies [1].

III. System model

In this section, we introduce the system model with the related assumptions that will be used to formulate an optimization problem for content allocation to vehicles minimizing the traffic downloaded from the cellular infrastructure.

A. Video Streaming Model

We consider a network with three types of nodes:

- *Infrastructure nodes* ($\mathcal{I}$). Base stations or macro-cells. They provide full coverage, and can serve any content request.
- *Helper nodes* ($\mathcal{H}$). Vehicles such as cars, buses, taxis, trucks, etc., where $|\mathcal{H}| = h$. These are used to store popular content and to serve user requests at low cost through a direct vehicle to mobile node link.
- *End user nodes* ($\mathcal{U}$). Mobile devices such as smartphones, tablets or netbooks. These nodes request (non-live) video content for streaming to $\mathcal{H}$ and $\mathcal{I}$ nodes.

Each video consists of a number of small chunks that are downloaded into a $\mathcal{U}$ node's playout buffer in order, and consumed for playout as follows:

Helper download. When a $\mathcal{U}$ node is in range of (at least) an $\mathcal{H}$ node that stores the requested content, the next immediate chunks not yet in the playout buffer are downloaded at low cost *in order* at mean rate $\mathbf{E}[r_H]$. This mean rate can be easily inferred from mobility statistics (distribution of number of caches during a contact) and download rate distribution, and it depends on the cell association policy used. E.g., assume that a node is currently viewing chunk n, and its playout buffer already contains chunks $n+1, \ldots, n+k$; then, chunks starting from chunk $n+k+1$ will be downloaded until the connection with that $\mathcal{H}$ node is lost. This opportunistic connection is represented by the green region in Figure 1 (e.g., between t_1 and t_2 the node will download from cache 1). What is more, we do not allow for simultaneous connections, i.e., a $\mathcal{U}$ node can download from at most one $\mathcal{H}$ node at a time (we defer considering multi-connectivity to future work). For this reason, in Figure 1 the user will switch to cache 3 only at t_4, i.e., after it has finished downloading from cache 2. If multiple caches are available, the user chooses what the helper to connect depending on a predefined policy (e.g., max SINR).

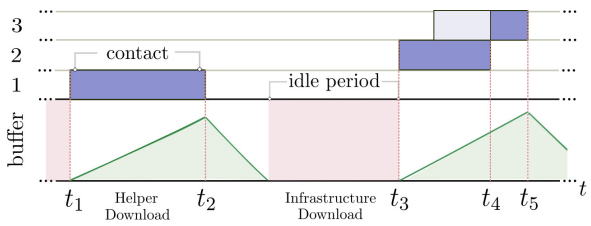


Fig. 1: Sequence of contacts with three caches (above), and amount of data in end user buffer over time (below, in green). The red region indicates when data is downloaded from $\mathcal{I}$.

Infrastructure download. When a $\mathcal{U}$ node is not in range of an $\mathcal{H}$ node that stores the requested content *and* its playout buffer is (almost) empty, new chunks are downloaded from the infrastructure at a mean rate $\mathbf{E}[r_I]$ until another $\mathcal{H}$ node storing the content is encountered. The communication between $\mathcal{U}$ and $\mathcal{I}$ nodes has a high cost in terms of energy consumption [28] and bandwidth of the backhaul links [32]. The download from the infrastructure corresponds to the red region of Figure 1. However, if the playout buffer is *not* empty, no chunks are downloaded from $\mathcal{I}$ until the buffer empties.

Playout. Chunks in the playout buffer are consumed at a mean viewing *playout* rate $\mathbf{E}[r_P]$. We only require the buffer to be large enough to fit the requested file which is a reasonable assumption in modern devices.

B. Main Assumptions

A.1 - Catalogue. Let $\mathcal{K}$ be the set of all possible contents that users might request (also defined as "catalogue") where $|\mathcal{K}| = k$. Let further c be the size of the cache in each vehicle. We make the natural assumption that $c \ll k$. A content $i \in \mathcal{K}$ is of size s_i, and is characterized by a popularity value ϕ_i measured as the expected request rate within a seeding time window from all users and all cells. Similar to a number of works on edge caching [10], [25], we assume this time window to be a system parameter chosen by the MNO. Every time window, the MNO refreshes its caches installed in vehicles according to the new estimated popularity. However, while it is reasonable to assume the content size is known, predicting the popularity of a content is more challenging. Nevertheless, several studies have confirmed that simple statistical models (e.g., ARMA) along with content characteristics can help to have good estimation of the request rate, at least in the immediate future [33]. Without loss of generality, we assume content is sorted by decreasing popularity as $\phi_1 \geq \phi_2 \geq \ldots \geq \phi_k$.

A.2 - Mobility model. We assume that the inter-meeting times $T_i^{(w)}$ between a user requesting content $i \in \mathcal{K}$ and a vehicle $w \in \mathcal{H}$ are IID random variables characterized by a known *generic* distribution $f_C(t)$ with mean rate λ. Although

previous work shows that urban settings can reveal different mobility classes (i.e., different λ), such differences are less evident at a finer granularity (sub-areas of a city). Since during the video playout there is little chance for a $\mathcal{U}$ node to move from an area to another, we believe that considering a single mobility class is a good approximation for the scenario considered. Contact durations are drawn from another *generic* distribution $f_D(t)$ with mean $\mathbf{E}[D]$. The mean contact duration is calculated by considering the association time due to the switching between nodes. We assume f_C and f_D to have bounded first and second moments. Finally, we denote with T_i the inter-meeting times between a user requesting content $i \in \mathcal{K}$ and *any* vehicle storing such a content.

A.3 - Cache model. Let $x_i^{(w)} \in \{0,1\},\ i \in \mathcal{K}, w \in \mathcal{H}$ be an indicator variable denoting if helper node w stores content i. We assume $\mathcal{H}$ nodes to *store the whole content*, i.e., fractional storage is not allowed (we will relax this assumption later in the paper). Let further x_i denote the number of $\mathcal{H}$ nodes storing content i, i.e., $x_i \triangleq \sum_{w\in\mathcal{H}} x_i^{(w)}$. The vector $\mathbf{x}$ will be the control variable for our optimal cache allocation problem.

A.4 - Content download rate. We assume that the mean download rate from the infrstructure $\mathbf{E}[r_I]$ is larger than the mean playout rate of videos[3] $\mathbf{E}[r_P]$. When this assumption is not true, there might be stalls during the playout which would have occurred independently of our vehicular cloud. We further assume $\mathbf{E}[r_H]$ to be larger than $\mathbf{E}[r_P]$ which is reasonable due to the reduced distance between users and helper nodes. Scenarios where $\mathbf{E}[r_I]$ (and/or $\mathbf{E}[r_H]$) are lower than the playout rate require initial buffering which significantly degrades QoE [14]. Nevertheless, our framework could be easily extended to include such an initial buffering. In this paper, we set $\mathbf{E}[r_I] = \mathbf{E}[r_P] + \epsilon$ ($\epsilon > 0$ small) to limit the access to the cellular infrastructure to the minimum required rate to ensure smooth playout as in most modern media players (e.g., YouTube). For simplicity, we assume ϵ equal to 0.

A.5 - Data offloading. A request for content i will download a number of bytes from $\mathcal{I}$ nodes. This number is a random variable W_i that depends on x_i as well as the sample path of the contact variables. We denote as $\mathbf{E}[W_i|x_i]$ the expected value of this quantity per content where the expectation depends on $f_C(t)$ and $f_D(t)$. Our goal is to minimize it, since $s_i - \mathbf{E}[W_i|x_i]$ is the traffic *offloaded* on average for requests of content i. To keep notation simple, we will refer to this quantity as $\mathbf{E}[W_i]$.

Table I shows the main notation used in the paper.

TABLE I: Main notation used in the paper.

CONTROL VARIABLES AND SETS	
x_i	Number of replicas stored for content i
X	Feasible region for $\mathbf{x}$
$\mathcal{I}$	Infrastructure nodes
$\mathcal{H}$	Helper nodes
$\mathcal{U}$	End user nodes
$\mathcal{K}$	Content catalogue
CONTENT	
k	Number of content in the catalogue
ϕ_i	Request rate for content i
θ_{ij}	Internal chunk popularity at chunk j for content i
s_i	Size of content i
c	Buffer size per vehicle
MOBILITY	
λ	Mean inter-meeting rate between $\mathcal{U}$ and $\mathcal{H}$ nodes
$\mathbf{E}[D]$	Mean contact duration
h	Number of vehicles
CHUNK DOWNLOAD	
W_i	Bytes downloaded for content i from $\mathcal{I}$ nodes
Ψ_{ij}	Probability to offload chunk j for content i before its playout
$\mathbf{E}[r_P]$	Mean viewing playout rate
$\mathbf{E}[r_I]$	Mean download rate from $\mathcal{I}$ nodes (equal to r_P)
$\mathbf{E}[r_H]$	Mean download rate from $\mathcal{H}$ nodes
$\mathbf{E}[Y_i]$	Expected bulk size

IV. Optimal per-content allocation

In Section IV-A, we first formulate an optimization problem to minimize the traffic downloaded from the cellular infrastructure. Then, we approximate analytically the mean number of bytes downloaded from $\mathcal{H}$ nodes for two regimes of mobile vehicle density, and we solve the related problem to find the optimal content allocation. Specifically, in Section IV-B we consider first a *low vehicle density* scenario which provides insights on how to solve the *generic vehicle density* scenario of Section IV-C. Finally, in Section IV-D we provide an analytical bound on the approximation error introduced.

[3]We are only interested in the mean data rates since our model performs stationary regime analysis as we will show in the next sections.

A. Offloading Optimization Problem

Given the above assumptions, we can propose a policy where: (i) the user's video is never interrupted provided the infrastructure can guarantee at least the playout rate; (ii) while the video plays out at the user, future parts of it are actually downloaded from locally encountered caches (in principle pre-fetched) thus offloading some of it from the infrastructure. As long as the playout buffer remains non-empty, $\mathcal{I}$ nodes never need to be accessed. And when they do, we ensure that the minimum necessary amount of bytes is downloaded from the infrastructure ($\mathbf{E}[r_I] = \mathbf{E}[r_P] + \epsilon$). The goal of the operator is to minimize the amount of bytes downloaded per content $\mathbf{E}[W_i]$, among all content $i \in \mathcal{K}$, by appropriately choosing the control variable $\mathbf{x}$. This is captured in the following:

Problem 1. *Consider the video streaming model above. The solution to the following problem minimizes the expected number of bytes downloaded from the cellular infrastructure:*

$$\begin{aligned} &\underset{\mathbf{x}\in X^k}{\textit{minimize}} && \sum_{i=1}^{k} \phi_i \cdot \mathbf{E}[W_i], \\ &\textit{subject to} && \sum_{i\in\mathcal{K}} s_i \cdot x_i^{(w)} \le c, \quad \forall w \in \mathcal{H}, \end{aligned} \tag{1}$$

where $X \triangleq \{a \in \mathbb{N} \mid 0 \le a \le h\}$ is the feasible region for the control variable $\mathbf{x}$, and $\mathbf{E}[W_i]$ is the expected number of bytes downloaded from $\mathcal{I}$ for content i when x_i $\mathcal{H}$ nodes store i.

The objective function counts the number of bytes downloaded from the infrastructure in a time window for the entire

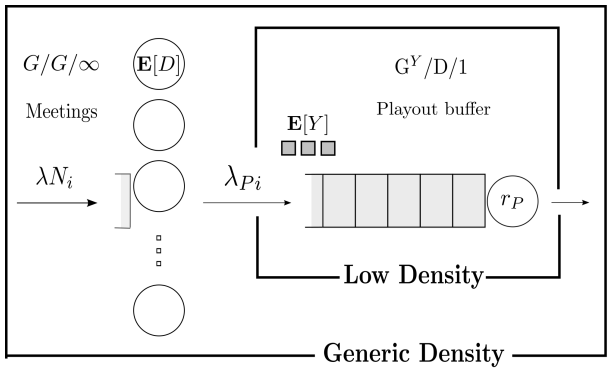


Fig. 2: Proposed queuing model for the playout buffer.

catalogue. For each content, this is equivalent to the content popularity times the expected number of bytes that cannot be offloaded through helper nodes. In the limit of many content requests during a time window, the expected value of W_i becomes asymptotically exact for a specific instance of requests. Note that, given the assumption of IID mobility, it suffices to optimize the total number of copies x_i without considering the per vehicle variables $x_i^{(w)}$ any more. Furthermore, the optimization problem is subject to the following constraints: (i) the number of replicas of content i cannot be negative; (ii) the number of replicas of content i cannot be higher than the number of vehicles participating in the cloud; (iii) each vehicle has a storage constraint and cannot store more than c bytes.

B. Content Caching with Low Density Model

First, let us assume that contacts with vehicles are sparse, i.e., the probability of overlapping contacts in time, with different vehicles *both storing the same content*, is small:

Definition 4.1. *We refer to Low Density Model when*

$$\lambda \cdot h \cdot \mathbf{E}[D] \ll 1 \qquad \textit{and} \qquad \frac{\mathbf{E}[r_H]}{\mathbf{E}[r_P]} < \frac{1}{\lambda \cdot h \cdot \mathbf{E}[D]}.$$

This is a reasonable scenario when the number of vehicles utilized in the cloud is small, or for low popularity content that do not have replicas in every vehicle. In this case, we model the playout buffer as a bulk $\mathrm{G}^Y/\mathrm{D}/1$ queue where new bytes arrive in *bulks* when a helper node with the requested content is encountered, and are consumed at a constant playout rate. This system is depicted inside the small square of Figure 2 (the queue on the left can be ignored for now). The following lemma holds:

Lemma 4.2. *Consider the Low Density Model. The following is asymptotically tight as the content size becomes large, i.e.,*

$$\lim_{s_i \to +\infty} \left[\mathbf{E}[W_i] - s_i \cdot \left(1 - \lambda \cdot x_i \cdot \mathbf{E}[D] \cdot \frac{\mathbf{E}[r_H]}{\mathbf{E}[r_P]} \right) \right] = 0.$$

Proof. Consider a content i currently stored in x_i caches. The $\mathrm{G}^Y/\mathrm{D}/1$ queue model for the playout buffer has:

- *Service Rate.* Jobs (i.e., bytes) in the buffer are served (i.e., viewed by the user) at the mean playout rate $\mathbf{E}[r_P]$.
- *Bulk Size.* A contact between a device and a helper node storing video i corresponds to a new (bulk) arrival in the playout buffer of the device. A new arrival brings a random amount of new bytes that depends on the contact duration with that $\mathcal{H}$ node. We denote the expected bulk size in bytes as $\mathbf{E}[Y] \triangleq \mathbf{E}[D] \cdot \mathbf{E}[r_H]$.
- *Arrival Rate.* The total arrival rate into the playout queue is $\lambda_{Pi} \triangleq \lambda \cdot x_i$ since there are x_i replicas of content i.

The long term utilization of the playout queue is

$$\rho_i = \lambda_{Pi} \cdot \frac{\mathbf{E}[Y]}{\mathbf{E}[r_P]}, \lambda \cdot x_i \cdot \mathbf{E}[D] \cdot \frac{\mathbf{E}[r_H]}{\mathbf{E}[r_P]}. \tag{2}$$

by Little's law. The necessary condition for this queue to be stable and ergodic is that $\rho_i < 1$, for any $i \in \mathcal{K}$. This condition is satisfied since $\lambda_{Pi} \leq \lambda \cdot x_i$ and Definition 4.1 applies.

Let $B_i^{(n)}$ (resp. $I_i^{(n)}$) be the length of the n^{th} busy (resp. idle) period of the playout buffer for content i. When the queue is stable, $\{(I_i^{(n)}, B_i^{(n)}), n \geq 1\}$ forms an alternating renewal process (as the queue regenerates at the end of each busy period). Let further $I_i^{(n)} + B_i^{(n)}$ define a cycle, and $P_I(t)$ the probability that the playout buffer is empty at time t. Since $\mathbf{E}[I_i^{(n)} + B_i^{(n)}] < \infty$ (by stability), it holds that

$$\lim_{t \to +\infty} P_I(t) = \frac{\mathbf{E}[I_i]}{\mathbf{E}[I_i] + \mathbf{E}[B_i]} = 1 - \rho_i, \tag{3}$$

where the second equality holds by ergodicity [27]. Let further associate to each cycle a reward equal to the bytes downloaded from the cellular infrastructure during that cycle, i.e., the reward in cycle n is equal to $I_i^{(n)} \cdot \mathbf{E}[r_P]$ (we remind the reader that the download rate from the infrastructure is assumed to be equal to the playout rate $\mathbf{E}[r_P]$, see Assumption A.4). Consider now a video of duration T_i and remember that W_i is equal to the number of total bytes downloaded from the infrastructure (see Assumption A.5). From the renewal-reward theorem (e.g., see Theorem 3.6.1 in Ross [27]) we have that

$$\lim_{T_i \to +\infty} \frac{\mathbf{E}[W_i]}{T_i} = \frac{\mathbf{E}[I_i] \cdot \mathbf{E}[r_P]}{\mathbf{E}[B_i] + \mathbf{E}[I_i]}.$$

We conclude our proof by combining Eq. (2), Eq. (3) and the fact that the duration $T_i = \frac{s_i}{\mathbf{E}[r_P]}$ for large s_i. □

The above results states that as s_i becomes large, we can easily express $\mathbf{E}[W_i]$ in closed form. We will use this result as an approximation for finite size content to introduce the optimal allocation problem for the Low Density Model. Later, we elaborate on the error introduced for small content.

Problem 2. *Consider the Low Density Model. The solution to the following optimization problem minimizes the expected number of bytes downloaded from the cellular infrastructure:*

$$\begin{aligned} \underset{\mathbf{x} \in X^k}{\textit{maximize}} \quad & \sum_{i=1}^{k} \phi_i \cdot s_i \cdot x_i, \\ \textit{subject to} \quad & \sum_{i \in \mathcal{K}} s_i \cdot x_{ij} \leq c, \quad \forall j \in \mathcal{H}. \end{aligned} \tag{4}$$

Proof. Our objective is to minimize the total number of bytes downloaded from $\mathcal{I}$ nodes. Based on Lemma 4.2, and simplifying the constant factors, we obtain Eq. (4). □

Proposition 4.3. *Problem* (2) *is NP-hard.*

Proof. The problem is a bounded knapsack problem (BKP) with "profit" $\phi_i \cdot s_i$ and "cost" s_i for element i with *individual* capacity constraints (instead of having one single *global* capacity). BKP is NP-hard [23]. □

The above proposition states that Problem (2) is hard due to the integer nature of variables x_i (by reduction to a BKP). Similar to a number of works, we consider here the *continuous relaxation* of the problem. In this case, the continuous relaxation brings two fundamental advantages: first, the problem may be solved in polynomial time whereas the classic BKP is NP-hard; second, it is possible to evaluate the quality of a feasible set of solutions. What is more, if the number of replicas is fractional, we can replace the *individual* capacity constraint of Eq. (1) with a *global* capacity constraint, i.e., $\sum_{i\in\mathcal{K}} s_i \cdot x_i \leq c \cdot h$. It is easy to see that, if $\mathbf{x}$ is fractional, any allocation that fits the global capacity is also a feasible allocation[4]. In other words, we solve now an optimization problem that replace Eq. (1) with the global capacity constraint and $\mathbf{x}$ is real. This new problem, that is equivalent to Problem (2) in the continuous case, is called *continuous BKP*. An optimal solution for such a problem can be determined by ordering items according to non-increasing cost per unit weight. More precisely, we can prove the following result:

Theorem 4.4. *The solution of Problem* (2) *under a continuous relaxation of* $\mathbf{x}$ *is given by*

$$x_i^* = \begin{cases} h, & \text{if } i < \gamma, \\ h \cdot \left(c - \sum_{i=1}^{\gamma-1} s_i\right) / s_\gamma, & \text{if } i = \gamma, \\ 0, & \text{if } i > \gamma, \end{cases}$$

where $\mathbf{x}^* \triangleq \arg\max_{\mathbf{x}\in\hat{X}^k} \sum_{i=1}^{k} \phi_i \cdot s_i \cdot x_i$*, and* γ *is the maximum content index such that* $\sum_{i=1}^{\gamma-1} s_i \leq c$.

Proof. We convert the problem into a 0-1 knapsack problem using a standard transformation. We remind the reader that content is ordered in decreasing popularity (i.e., $\phi_1 \geq \phi_2 \geq \cdots \geq \phi_k$). For each content i, we create h different "virtual" contents with $\{profit, cost\}$ equal to

$$\{0,0\}, \{\phi_i \cdot s_i, s_i\}, \{2 \cdot \phi_i \cdot s_i, 2 \cdot s_i\}, \ldots, \{h \cdot \phi_i \cdot s_i, h \cdot s_i\},$$

that gives a total of $h \cdot k$ total elements (instead of the original k). We can then consider content ordered by profit per unit weight, i.e., $\frac{\text{profit}}{\text{cost}} = \frac{\phi_i \cdot s_i}{s_i} = \phi_i$. Note that all these elements have the same profit per bit (so, for content i we can always pick the respective highest allocation $\{h \cdot \phi_i \cdot s_i, h \cdot s_i\}$, i.e., storing that content in all helpers). If all items can be fully included in each cache (i.e., $\sum_{i\in\mathcal{K}} s_i \leq c$), then any optimal solution will fully contain all items. Otherwise, when $\sum_{i\in\mathcal{K}} s_i > c$, there exists only one item $\gamma \in \{1, \ldots, k\}$ such that $\sum_{i=1}^{\gamma-1} s_i \leq c$ and $\sum_{i=1}^{\gamma} s_i > c$. Now consider the greedy allocation based on profit per bit (which turns out to be the content popularity ϕ_i in our case) proposed by the theorem. Kellerer *et al.* [17] show that this greedy allocation is also an optimal solution to the continuous BKP. □

[4]We define *feasible* an allocation that satisfies the individual capacity constraint of Eq. (1).

TABLE II: Efficiency of Algorithm 1 for different cache sizes.

Cache size	0, 01%	0, 05%	0, 10%	0, 50%
Algorithm 1	99,947%	99,994%	99,997%	100,00%

Corollary 4.5. *The continuous relaxation of Problem* (2) *can be optimally solved in time* $O(k)$.

Proof. Because of the content sorting, an optimal solution would normally take time $O(k \log k)$. However, an algorithm to find weighted medians [19] solves it in time $O(k)$. □

Theorem 4.4 optimally solves Problem (2) after the continuous relaxation. However, this optimal allocation is feasible only when fractional storage is allowed. We propose a greedy algorithm (Algorithm 1) based on Theorem 4.4 to infer a feasible integer allocation. In this algorithm, the $\gamma - 1$ most popular contents are replicated in every vehicle. Then, the remaining storage *per vehicle* is filled greedily with contents, sorted by popularity, that fit in the cache, if any. When caches are large compared to the average content size, the error introduced by this greedy solution is expected to be low as we have verified through numerical simulations: in Table II, we compare the *efficiency* of Algorithm 1 for different content size. We define as efficiency of an allocation policy the traffic offloaded by the integer allocation (provided by Algorithm 1 in this case) over the traffic offloaded by the related continuous relaxation (Thereom 4.4). As the latter is a lower bound on the optimal solution of the discrete problem, the actual performance gap is bounded by the real-valued solution.

Algorithm 1 Caching Algorithm for Low Density Model

Input: $\mathbf{s}$, ϕ, c, h, k
$\mathbf{x} \leftarrow \emptyset$
j$\leftarrow 1$
while $\sum_{i=1}^{j} s_i \leq$ c **do**
  $x_j \leftarrow$ h
  j $\leftarrow$ j + 1
w$\leftarrow \sum_{i=1}^{j-1} s_i$
for i$\leftarrow$ j to k **do**
  if w + $s_j \leq$ c **then**
    $x_i \leftarrow$ h
    w + s_j
return x

We have seen that an optimal allocation tends to replicate the most popular content in every vehicle. This finding is interesting because, even if contacts do not overlap, *a node still sees multiple caches during the playout of a content*. One would expect that these caches should store different content to maximize diversity. E.g., in the femto-caching setup, storing the most popular content in all caches is optimal only when caches are isolated, but it is suboptimal when a node has access to multiple caches [11]. We will see that this most popular allocation is no longer efficient when vehicle density increases. Beyond providing performance guarantees, this policy is in line with the standard caching policies in single caches which store the most popular content [11].

C. Content Caching with Generic Density Model

We now consider the following *busy* urban environment:

Definition 4.6. *We refer to Generic Density Model as a scenario where contacts with different vehicles (with the same content) might overlap, i.e., $\lambda \cdot x_i \cdot \mathbf{E}[D]$ is not small.*

If a user is downloading video i from node A, and the connection is lost (e.g., user or cache moves away), the user could just keep downloading from another node B storing i, also in range. Hence, as long as there is *at least* one cache with a copy within range (we denote this time interval with B_i), the user will keep downloading content i at rate $\mathbf{E}[r_H]$[5]. We cannot apply the previous model directly, because when the user above switches from cache A to cache B, the contact with cache B might be already ongoing, and we are interested in the *residual* contact duration with B, which is generally different (unless contact durations are exponential). We can then model these overlapping contacts with an extra G/G/∞ queue in front of the playout queue (as shown in Figure 2). New vehicles arrive in the G/G/∞ queue with rate $\lambda \cdot x_i$, each staying for a random service time (corresponding to a contact duration with mean $\mathbf{E}[D]$) and independently of other cars. The number of jobs in the G/G/∞ queue is the number of vehicles concurrently within range of the user.

Hence, it is easy to see that: (i) the beginnings of busy periods of the queue on the left of Figure 2 correspond to new bulk arrivals in the playout buffer (queue on the right), and (ii) the mean duration of such busy periods, multiplied by $\mathbf{E}[r_H]$, corresponds to the (new) mean bulk size per arrival[6].

Lemma 4.7. *Consider the Generic Density Model. If x_i large and λ small, the bulk arrival statistics in the playout buffer are*

$$\lambda_{Pi} = \lambda \cdot x_i \cdot e^{-\lambda \cdot \mathbf{E}[D] \cdot x_i}, \tag{5}$$

$$\mathbf{E}[Y_i] = \frac{\mathbf{E}[r_H]}{\lambda \cdot x_i} \cdot \left(e^{\lambda \cdot \mathbf{E}[D] \cdot x_i} - 1\right). \tag{6}$$

Proof. Let $\Gamma_i^{(w)}(t)$ be a point process with rate λ. Each point of this process is the time of a contact between a user and a vehicle $w \in \mathcal{H}$ storing content $i \in \mathcal{K}$. The inter-arrival time between two points is captured by the random variable $T_i^{(w)}$ (see Assumption A.2). To clarify, assume $T_{i(1)}^{(w)}$ to be a random variable that corresponds to the time of the first jump. Then, the event $\{\Gamma_i^{(w)}(t) = 0\}$ is equivalent to the event $\{T_{i(1)}^{(w)} > t\}$, meaning that the first jump will occur after epoch t, i.e.,

$$\mathbf{P}[T_{i(1)}^{(w)} > t] = \mathbf{P}[\Gamma_i^{(w)}(t) = 0].$$

Since content i has x_i replicas, there are x_i identical processes $\Gamma_i^{(w)}(t)$. We equivalently redefine $\Gamma_i^{(w)}(t)$ as follows: $\{\Gamma_i^{(w)}(t), t > 0, w \in \mathcal{H} \mid x_i^{(w)} = 1\}$ are x_i identical and independent renewal processes with *holding times* $T_i^{(w)}$ corresponding to the inter-arrival times between users and vehicles storing content i. Let further $\{\Gamma_i(t), t > 0\}$ be the superposition of these processes. According to the Palm-Khintchine theorem [16], $\{\Gamma_i(t), t > 0\}$ approaches a Poisson process with rate $\lambda \cdot x_i$ if x_i large and λ small. Note that T_i corresponds to the inter-arrival times of the process $\{\Gamma_i(t), t > 0\}$ (assumption A.2). Thus, T_i approaches an exponential distribution[7] with mean rate $\lambda \cdot x_i$, and the G/G/∞ queue capturing overlapping meetings (Figure 2 left) can be approximated by an M/G/∞ queue with arrival rate $\lambda \cdot x_i$ and mean service time $\mathbf{E}[D]$.

The probability that there are 0 jobs in the system (idle probability) is $e^{-\lambda \cdot \mathbf{E}[D] \cdot x_i}$ (this result is well known for M/M/∞ queue, but it also holds for generic contact durations by the insensitivity of the M/G/∞ queue [13]). Furthermore, by ergodicity, it holds that[8] $\frac{\mathbf{E}[I_i]}{\mathbf{E}[B_i]+\mathbf{E}[I_i]} = e^{-\lambda \cdot \mathbf{E}[D] \cdot x_i}$. Since $\mathbf{E}[I_i] = \frac{1}{\lambda \cdot x_i}$, solving for $\mathbf{E}[B_i]$ gives us the expected busy period of the M/G/∞ queue and multiplying by $\mathbf{E}[r_H]$ gives as the expected bulk size $\mathbf{E}[Y_i]$ of Eq. (6).

Additionally, the beginnings of busy periods of the M/G/∞ queue correspond to (bulk) arrivals into the playout queue. The mean time between such arrivals is simply $\mathbf{E}[B_i] + \mathbf{E}[I_i]$. Hence, the arrival rate of bulks into the playout buffer is

$$\lambda_{Pi} \triangleq \frac{1}{\mathbf{E}[B_i] + \mathbf{E}[I_i]} = \frac{1}{\frac{e^{\lambda \cdot \mathbf{E}[D] \cdot x_i} - 1}{\lambda \cdot x_i} + \frac{1}{\lambda \cdot x_i}} = \lambda \cdot x_i \cdot e^{-\lambda \cdot \mathbf{E}[D] \cdot x_i}.$$

□

Lemma 4.8. *Consider the Generic Density Model. The following expression is asymptotically tight as the content size s_i becomes large, when $x_i < \frac{1}{\lambda \cdot \mathbf{E}[D]} \cdot \ln\left(\frac{\mathbf{E}[r_H]}{\mathbf{E}[r_H]-\mathbf{E}[r_P]}\right)$:*

$$\lim_{s_i \to \infty} \left[\mathbf{E}[W_i] - s_i \cdot \left[1 - \left(1 - e^{-\lambda \cdot \mathbf{E}[D] \cdot x_i}\right) \cdot \frac{\mathbf{E}[r_H]}{\mathbf{E}[r_P]}\right]\right] = 0. \tag{7}$$

Proof. $\mathbf{E}[Y_i]$ corresponds now to the expected bulk size for an arrival in the playout buffer (instead of $\mathbf{E}[D] \cdot \mathbf{E}[r_H]$ in the low density model). Similarly to Lemma 4.2, we multiply the input rate λ_{P_i} of Eq. (5) with the bulk size $\mathbf{E}[Y_i]$ of Eq. (6), and divide by the playout rate $\mathbf{E}[r_P]$, to obtain the utilization of the playout buffer in the generic case:

$$\rho_i = \left(1 - e^{-\lambda \cdot \mathbf{E}[D] \cdot x_i}\right) \cdot \frac{\mathbf{E}[r_H]}{\mathbf{E}[r_P]}.$$

From this point on, we can follow the exact steps of the proof of Lemma 4.2, using this new ρ_i, to get the desired Eq. (7). Note, however, that unlike the Low Density Model, here the utilization of the playout buffer is lower than one when

$$x_i \geq \frac{1}{\lambda \cdot \mathbf{E}[D]} \cdot \ln\left(\frac{\mathbf{E}[r_H]}{\mathbf{E}[r_H] - \mathbf{E}[r_P]}\right) \triangleq \hat{h}, \tag{8}$$

where $\hat{h}$ is an upper bound on the allocation. Otherwise the queue is not stationary. Physically, this essentially means that the delivery capacity of the helper system is much higher

[5] We ignore for now interruptions from switching between nodes which can be very small if vehicles are operating as LTE relays [31]. We consider switching and association delays in the simulations.

[6] Note that the new bulk size $\mathbf{E}[Y_i]$ depends on x_i.

[7] We invite the reader to note that inter-arrival times approaches an exponential distribution if x_i large and λ small which was an assumption *not* needed for the Low Density Model. However, while this assumption (i.e., x_i large) might not always be true, exponential inter-meeting times have been largely used in literature and considered as a good approximation, especially in the tail of the distribution [8], [15].

[8] We slightly abuse notation for these idle and busy periods of the queue on the left, while in the proof of Lemma 4.2 we used them for the idle and busy period of the playout buffer (queue on the right).

than $\mathbf{E}[r_P]$, and (for long enough content) the infrastructure is not needed. In theory, this would make the playout queue non-stationary. In practice, this implies that we have allocated too many copies for this content, at least in our model. We will therefore use Eq. (8) as an additional constraint in the allocation problem. □

We can now formulate the optimal cache allocation problem for the Generic Density Model:

Problem 3. *Consider the Generic Density Model. The solution to the following optimization problem minimizes the expected number of bytes downloaded from the cellular infrastructure:*

$$\begin{aligned} &\underset{\mathbf{x}\in\hat{X}^k}{\textit{maximize}} \quad \Phi(\mathbf{x}) \triangleq \sum_{i=1}^{k} \phi_i \cdot s_i \cdot e^{-\lambda\cdot\mathbf{E}[D]\cdot x_i}, \\ &\textit{subject to} \quad \sum_{i\in\mathcal{K}} s_i \cdot x_{ij} \le c, \quad \forall j \in \mathcal{H}. \end{aligned} \tag{9}$$

where $\hat{X} \triangleq \{a \in \mathbb{N} \mid 0 \le a \le \min\{h, \hat{h}\}\}$ *is the feasible region for the control variable* $\mathbf{x}$*.*

Proof. Based on Lemma 4.8, and simplifying the constant factors, we obtain Eq. (9). What is more, we upper bound the feasible region of the control variable $\mathbf{x}$ according to Eq. (8). □

Proposition 4.9. *Problem* (3) *is NP-hard.*

Proof. The problem corresponds to the a nonlinear BKP which is NP-hard. □

Branch-and-bound algorithms have been developed for such problems, but they are not efficient when the size of the problem increases. Instead, similar to the Low Density Model, we consider here the continuous relaxation of Problem (3). In the convex case, the continuous relaxation is itself convex, and therefore likely to be tractable:

Theorem 4.10. *The solution of Problem* (3) *is given by*

$$x_i^* = \begin{cases} 0, & \text{if } \phi_i < L, \\ \frac{1}{\lambda\cdot\mathbf{E}[D]} \cdot \ln\left(\frac{\lambda\cdot\mathbf{E}[D]\cdot\phi_i}{m_C}\right), & \text{if } L \le \phi_i \le U, \\ \min\{h, \hat{h}\}, & \text{if } \phi_i > U, \end{cases}$$

where $\mathbf{x}^* \triangleq \arg\max_{\mathbf{x}\in\hat{X}^k} \Phi(\mathbf{x})$*,* $L \triangleq \frac{m_C}{\lambda\cdot\mathbf{E}[D]}$ *and* $U \triangleq \frac{m_C}{\lambda\cdot\mathbf{E}[D]} \cdot e^{\min\{h,\hat{h}\}\cdot\lambda\cdot\mathbf{E}[D]}$*, and* m_C *is an appropriate Lagrange multiplier.*

Proof. Problem (3) is clearly convex since the objective function is a sum of convex functions, the constraints are linear and the domain of the feasible solutions is convex. We solve it by Karush-Kuhn-Tucker (KKT) conditions, i.e.:

$$\begin{cases} l_i \cdot x_i = 0 \\ m_i \cdot (h - x_i) = 0 \\ m_C \cdot \left(c \cdot h - \sum_{i=1}^{k} s_i \cdot x_i\right) = 0 \end{cases}$$

where l_i and m_i are appropriate Lagrange multipliers related to the bounds of $\mathbf{x}$. For such a problem, this method provides necessary and sufficient conditions for the stationary points to be optimal solutions. The Lagrangian function $\mathcal{L}(\mathbf{x})$ is

$$-\Phi(\mathbf{x}) + \sum_{i=1}^{k} [l_i \cdot x_i + m_i \cdot (h - x_i)] + m_C \cdot \left(c \cdot h - \sum_{i=1}^{k} s_i \cdot x_i\right).$$

We compute the stationary points by computing the derivative of the Lagrangian function for each content i. Since the problem is convex, these points are global solutions. Making explicit $\mathbf{x}$, we obtain

$$x_i = \frac{1}{\lambda\cdot\mathbf{E}[D]} \cdot \ln\left(\frac{\lambda\cdot\mathbf{E}[D]\cdot\phi_i}{m_C - l_i + m_i}\right).$$

Then, the system constraints create three regimes depending on the content popularity:

- *Low popularity*. The optimal allocation $\mathbf{x}$ must be greater or equal to 0. According to the KKT conditions, we have two cases that satisfy the constraint: (i) $x_i > 0,\ l_i = 0$; (ii) $x_i = 0,\ l_i > 0$. The threshold between case (i) and (ii) depends on the content popularity: specifically, a content will get more than 0 copies when its popularity is higher than L which can be easily computed when $x_i > 0$:

$$\frac{1}{\lambda\cdot\mathbf{E}[D]} \cdot \ln\left(\frac{\lambda\cdot\mathbf{E}[D]\cdot\phi_i}{m_C}\right) > 0 \ \Leftrightarrow\ \phi_i > \frac{m_C}{\lambda\cdot\mathbf{E}[D]} \triangleq L.$$

- *High popularity*. The content allocation is upper bounded by the number vehicles h participating in the cloud. However, we also consider the potential tighter upper-bound $\hat{h}$ on x_i due to the stability condition for the playout buffer according to the discussion in the proof of Lemma 4.8. Similarly to the previous scenario, due to the KKT conditions, the constraint is satisfied when: (i) $x_i < \min\{h, \hat{h}\},\ m_i = 0$; (ii) $x_i = \min\{h, \hat{h}\},\ m_i > 0$. Again, the threshold between case (i) and (ii) depends on the content popularity: specifically, a content will get less than $\min\{h, \hat{h}\}$ copies when its popularity is lower than U which can be easily computed when $x_i < \min\{h, \hat{h}\}$:

$$\phi_i < \frac{m_C \cdot e^{\min\{h,\hat{h}\}\cdot\lambda\cdot\mathbf{E}[D]}}{\lambda\cdot\mathbf{E}[D]} \triangleq U.$$

- *Medium popularity*. In all the other cases (i.e., when $U \le \phi_i \le L$), the optimal allocation is proportional to the logarithm of the content popularity.

□

We use *randomized rounding* [26] on the content allocation of Theorem 4.10 to go back to a feasible integer allocation (and deal with the individual capacity constraint) which is a widely used approach for designing and analyzing such approximation algorithms. As argued earlier, the expected error is small when caches fit several contents. To validate this, we perform numerical simulation to calculate the efficiency[9] of the integer solution compared to the continuous one of Theorem 4.10 (Table III).

[9]The definition of efficiency is given at the end of Section IV-B.

TABLE III: Efficiency of the content allocation of Theorem 4.10 after randomized rounding for different cache sizes (in percentage of the catalogue size).

Cache size	0, 02%	0, 05%	0, 10%	0, 20%
Rounded	99,903%	99,975%	99,993%	99,998%

D. Non-stationary Playout Buffer

In order to calculate the number of bytes downloaded from the infrastructure, we have assumed that a video sees a stationary regime, regardless of its size. In practice, video files have finite sizes. In the next theorem, we show that the predicted amount of such bytes calculated with the stationary assumption is in fact a lower bound on the actual number of bytes, which is only asymptotically tight. We also analytically derive the exact estimation error as a function of content size and scenario parameters. This quantity could perhaps be used to derive an even better estimate for the objective of our problem, and further improve performance.

Proposition 4.11. *The following statements are true:*

1) *The stationary estimate* $\mathbf{E}[W_i]$ *is a lower bound on the expected amount of bytes downloaded from* $\mathcal{I}$ *nodes for any content size* s_i.
2) *The additional expected number of bytes downloaded as a function of* $\mathbf{x}$ *is*

$$\mathbf{E}[e_i] = \frac{\mathbf{E}[r_H] \cdot \mathbf{E}[{B_i}^2]}{2 \cdot (\mathbf{E}[B_i] + \mathbf{E}[I_i])}, \tag{10}$$

where $\mathbf{E}[B_i]$ *(resp.* $\mathbf{E}[I_i]$*) is the mean busy (resp. idle) period of the playout buffer for content* i *and* $\mathbf{E}[B_i^2]$ *is its second moment.*

Proof. 1) The lower bound can be proven using a sample path argument. Consider the stationary process $S(t)$, counting the number of bytes in the playout buffer, at time t, for a very long file that has started streaming at time $-\infty$. Consider now a finite size file request that starts streaming at some random time t_0, and denote its playout buffer size as $S^{'}(t)$. If the request arrives during an idle period of $S(t)$, then $S^{'}(t_0) = S(t_0) = 0$ and the two sample paths are the same (*coupled*) from that point on. However, if the request arrives during a busy period of playout buffer $S(t)$, then $S^{'}(t_0) = 0$ but $S(t_0) > 0$, by definition of a busy period. Hence, the stationary file will download fewer bytes from the infrastructure in the next idle period, as it already has some to consume.

2) The error in the stationary estimate thus comes if the video request arrives during a busy period of $S(t)$. By renewal theory, this occurs with probability $\frac{\mathbf{E}[B_i]}{\mathbf{E}[B_i]+\mathbf{E}[I_i]}$. Furthermore, conditional on this event, the expected amount of bytes in the stationary playout buffer (i.e., the expected value of $S(t_0)$) is equal to the *age* of that busy period multiplied by $\mathbf{E}[r_H]$. From renewal theory and the inspection paradox, it holds that the expected age is equal to the expected *excess time* $\mathbf{E}[B_e]$, which is equal to $\frac{\mathbf{E}[B_i^2]}{2\mathbf{E}[B_i]}$, where $\mathbf{E}[B_i]$ and $\mathbf{E}[B_i^2]$ are the first and second moments of the busy periods of the playout buffer G^Y/D/1. Putting everything together, the expected error is given by Eq. (10). □

Corollary 4.12. *Assume that bulk arrivals are exponentially distributed. Then, we have*

$$\mathbf{E}[B] = -\tilde{B}'(s)|_{s=0} = \frac{\mathbf{E}[D]}{1-\rho_i},$$
$$\mathbf{E}[B^2] = \tilde{B}''(s)|_{s=0} = \frac{\mathbf{E}[D]^2}{(1-\rho_i)^3},$$

where $\tilde{B}(s)$ *is the Laplace transform of the busy periods.*

Proof. For low traffic (i.e., when the probability to have more than one job in the queue at the same time is low), the busy periods of the $M/G/\infty$ are trivially exponentially distributed. What is more, Hall [12] shows that under conditions of heavy traffic, busy periods are very nearly exponentially distributed as well. With this assumption, the playout queue G^Y/D/1 has arrival rate λ_B and mean bulk size $\mathbf{E}[Y_i]$ which are given by Lemma 4.7. Note that the busy periods of this queue are statistically equivalent to those of an M/G/1 queue, with the same arrival rate and mean service requirement $\mathbf{E}[D] = \frac{\mathbf{E}[Y_i]}{\mathbf{E}[r_P]}$. The utilization of this queue is

$$\rho_i = \left(1 - e^{-\lambda \cdot \mathbf{E}[D] \cdot x_i}\right) \cdot \frac{\mathbf{E}[r_H]}{\mathbf{E}[r_P]}.$$

We can thus calculate $\mathbf{E}[B_i]$ and $\mathbf{E}[B_i^2]$ from that M/G/1 queue instead. We can derive the Laplace transform of the busy periods of an M/G/1 queue in recursive form [13] as

$$\tilde{B}(s) = \tilde{S}(s + \lambda_B - \lambda_B \cdot \tilde{B}(s)),$$

where $\tilde{S}(s)$ is the Laplace transform of the service time of the M/G/1 queue. While this recursion does not allow to invert $\tilde{B}(s)$, we can use it to calculate the moments. □

V. Per-chunk adaptation

In the previous section, we have made the simplifying assumption that either all chunks or no chunks of a content must be cached in a vehicle. Said otherwise, every chunk of a content must have the same number of replicas. Yet, two opposing "forces" call for a per-chunk optimization policy: (i) it is perhaps wasteful to cache too many of the early chunks as there might not be enough time to fetch these chunks anyway; (ii) if early chunks have higher popularity than later chunks, the former perhaps deserve more storage space. To this end, we propose a heuristic which adapts the optimal solution per-content to an internal allocation which depends on the aforementioned tradeoff. Specifically, we design an allocation policy which takes as input the optimal per-content allocation (of Section IV) and provides a per-chunk allocation.

First, we relax Assumptions A.1 and A.3 as follows[10]:

A.1 bis - Catalogue. A content $i \in \mathcal{K}$ consists of a number of small chunks s_i, and is characterized by a popularity value ϕ_i measured as the expected request rate within a seeding time window from all users and all cells. Further, each chunk has a popularity $\phi_i \cdot \theta_{ij}$ where $\theta_{ij} \in (0, 1]$ is the normalized internal popularity of chunk j for content i.

A.3 bis - Cache model. The number of replicas of chunk j for content i is indicated by $x_{ij} \in [0, h]$.

Now, consider content i. Its optimal *per-content* allocation is of x_i copies which corresponds to $s_i \cdot x_i$ chunks. While in

[10] If not explicitly stated otherwise, assumptions of Section III still hold.

the previous section we had $x_{ij} = x_i \ \forall j$, the goal of this section is to find the vector of chunk allocation $\{x_{ij}\}$ that further improves the volume of traffic offloaded. The idea is to reshuffle the chunks according to the internal content popularity (i.e., chunk popularity) and to the chunk delay (i.e., when the chunk will start to be played out). We summarize this idea in the following optimization problem:

Problem 4. *Let x_i be the number of replicas of content i. The following optimization problem provides the optimal chunk allocation given the internal popularity θ_{ij}:*

$$\begin{aligned} \underset{0\le x_{ij}\le h}{\text{maximize}} \quad & \sum_{j=1}^{s} \theta_{ij} \cdot \Psi_{ij}(x_{ij}), \\ \text{subject to} \quad & \sum_{j} x_{ij} = s \cdot x_i. \end{aligned}$$

where Ψ_{ij} is the probability to offload a given chunk j from helper nodes before its playout time.

The probability Ψ_{ij} depends on the number of replicas, on the content, on the mobility statistics and on the time at which a chunk is played out. Although calculating the correct probability is difficult due to these several dependencies, we will provide two approximations of Ψ_{ij} which will be used to compute the per-chunk allocation.

1) Example 1 (proportional): assume Ψ_{ij} depends (linearly) only on the number of replicas. Hence,

$$\Psi_{ij}(x_{ij}) = \frac{1}{h} \cdot x_{ij}.$$

When Ψ_{ij} is defined as above, Problem (4) corresponds to a bounded knapsack problem. In this case, the most popular chunks will have h replicas and the others 0.

2) Example 2 (infinite bandwidth): if a vehicle storing a subset of chunks for content i comes in range to the user, we assume that the user is able to download all the chunks of content i stored by vehicle. Note, we do not require that this is true in a real setup. We simply say that our policy will assume so, for simplicity (and thus might not always be optimal). This assumption becomes more accurate if for example there are many vehicles and many contents to ensure each one stores only few chunks per content, and of course depends also on the content size and download rate between $\mathcal{H}$ and $\mathcal{I}$ nodes. The probability Ψ_{ij} can be rewritten as follows:

$$\Psi_{ij}(x_{ij}) = 1 - \exp\{-\lambda \cdot (j-1) \cdot \tau \cdot x_{ij}\}, \tag{11}$$

where τ is the duration of each chunk in seconds.

Proof. We have seen in Lemma 4.7 that arrival of vehicles in the user communication range is characterized by a Poisson process if x_{ij} large and λ small. $\Psi_{ij}(x_{ij})$ follows from the probability that any of the vehicles having a copy of chunk j of content i is met within the time tolerance for this chunk which is equal to $(j-1)\cdot\tau$. The contacts between the user and a vehicle are assumed to follow a Poisson process of rate λ. □

Lemma 5.1. *Consider a continuous relaxation of Problem* (4)*. When Ψ_{ij} is described by Eq.* (11)*, the optimal solution is given by*

$$x_{ij} = \frac{1}{w_j} \cdot \ln\left(\frac{w_j}{m_{chunk}(x_i)} \cdot \theta_{ij}\right),$$

where $w_j \triangleq \lambda \cdot (j-1) \cdot \tau$ and $m_{chunk}(x_i)$ is an appropriate Lagrange multiplier. Furthermore, due to the box constraints, we assign 0 replicas to low popular chunks or h replicas to high popular chunks.

Proof. Problem (4) is clearly convex since the objective function is a sum of convex functions, the constraints are linear and the domain of the feasible solutions is convex. We solve it by Karush-Kuhn-Tucker (KKT) conditions. For such a problem, this method provides necessary and sufficient conditions for the stationary points to be optimal solutions. □

According to the above discussion, we summarize here the proposed two-step algorithm:

- *Step 1.* Compute a per-content allocation $\mathbf{x}$ assuming all chunks have same popularity. The replication factor is the result of the queueing model proposed in Section IV.
- *Step 2.* Step 1 suggests to store $s_i \cdot x_i$ chunks for content i in the vehicular cloud. Reshuffle these chunks according to chunk popularity and chunk delay as shown, e.g., in Lemma 5.1.

VI. Performance evaluation

In this section, we perform simulations based on real traces for vehicle mobility and content popularity to confirm the advantages of the vehicular cloud and to validate our theoretical results.

A. Simulation Setup

We build a MATLAB tool to simulate the requests for YouTube videos in the centre of San Francisco. We use the following traces:

Vehicle mobility. We use the Cabspotting trace [24] to simulate the vehicle behaviour; this trace records the GPS coordinates for 531 taxis in San Francisco for more than three weeks with granularity of one minute. To improve the accuracy of our simulations, we increase the granularity to ten seconds by linear interpolation. We also use this trace to extract the necessary mobility statistics for our model (λ and $\mathbf{E}[D]$) which are summarized in Table IV.

User mobility. We use synthetic traces based on SLAW mobility model [20] which is one of the most accurate models for human mobility: in fact, studies of human walk traces have discovered significant statistical patterns (e.g., truncated power-law distributions of flights, pause-times and inter-contact times, fractal way-points) which are captured by this model. SLAW is largely used in literature (e.g., in [39]).

Content. We infer the number of requests per day from a database with statistics for 100.000 YouTube videos [35]. The database includes static (e.g., title, author, duration) and dynamic information (e.g., daily and cumulative views, shares, comments). In order to increase the number of simulations and to provide sensitivity analysis for content size, buffer capacity and cache density, we limit the number of content to 10.000 selected randomly from the initial set.

Inline with the proposed protocols (e.g., 802.11p, LTE ProSe), we consider two maximum communication ranges between $\mathcal{U}$ and $\mathcal{H}$ nodes: 100 m (short range) or 200 m

TABLE IV: Mobility statistics

Range	λ	$\mathbf{E}[D]$
Short range	0,964 day^{-1}	31,23 s
Long range	2,83 day^{-1}	50,25 s

(long range). As most wireless protocols implement some *rate adaptation* mechanism, our simulator also varies the communication rate according to the distance between the user and the mobile helper she is downloading from (while we consider the average download rate in the model). Given current wireless rates in the 802.* family and that near future vehicles will probably carry high speed mobile access points, we use a *mean* $\mathbf{E}[r_H]$ = 5 Mbps. We also set $\mathbf{E}[r_P]$ = 1 Mbps, that approximates the streaming of a 720p video. Additionally, we implement an association setup mechanism according to Sesia *et al.* [31] that introduces a delay of two seconds to synchronize a UE with a vehicle (i.e., the download from a cache starts two seconds after the beginning of the contact). Finally, we set the cache size per node c in the range 0,02-0,5 percent of the total catalogue which is an assumption that has also been used in related work [10], [25] (we use 0,1 percent as a default value). Unless otherwise stated, the mean video length is one hour (i.e., mean content size equal to 450 MB).

The simulator works as follows: first, it generates a set of content requests concentrated at day-time; inter-arrival times between two requests are exponentially distributed according to the IRM model that is the de facto standard in the analysis of storage systems. Next, the simulator associates to each request the coordinates (and the mobility according to the SLAW model) of the user requesting the content. Then, it allocates content in caches according to different allocation policies. For each request, the simulator reproduces the playout of the video: the end user buffer will be opportunistically filled when at least one cache storing the requested video is inside the communication range, depending on the mobility traces. Finally, content requests are generated over a period of five days. Because of the large number of requests in the period considered, the confidence interval is too small to be distinguishable and hence is ignored in the following plots.

B. Per-content Caching Strategy Evaluation

We consider and compare the following allocation policies:

- *Optimal*. This policy solves Problem (3) with Generic Density Model. The policy is described in Section IV-C.
- *Most popular (MP)*. This policy solves Problem (2) with Low Density Model. The policy is described by Algorithm 1 in Section IV-B.
- *Least Recently Used (LRU)*. Starting from a random initial allocation, this policy discards the least recently used item when there is a cache miss. Unlike the above two policies, LRU keeps updating the cache content, and thus could incur higher traffic on the backhaul (from $\mathcal{I}$ to $\mathcal{H}$ nodes).
- *Random*. Content is randomly allocated in $\mathcal{H}$ nodes.

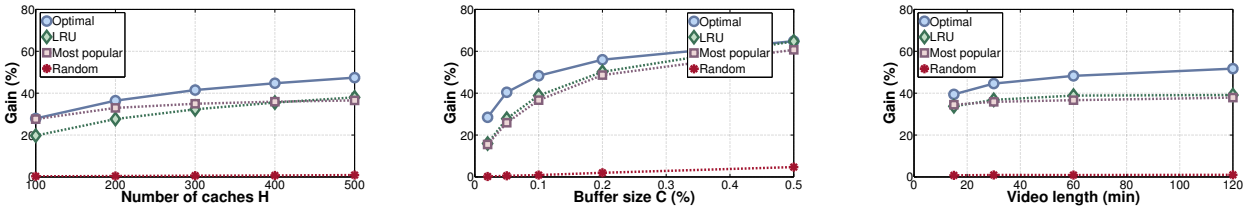
Cache Density. Figure 3 depicts the fraction of data offloaded by the vehicular cloud as a function of vehicle density, buffer capacity and video length, assuming long range communication. Specifically, in Figure 3a we vary the number of vehicles from 100 to 500. While the number of envisioned connected vehicles in the centre of San Francisco is expected to be much larger, it is really interesting to verify that a subset of them can still provide non-negligible offloading gains (more than 30 percent), which is important to promote the start up phase of the vehicular cloud. As proved in Section IV-B, the *MP* policy provides good performance in scenarios with low vehicle density: e.g., with 100 vehicles, the offloading gain is almost equal to the *optimal*. Conversely, for a larger number of vehicles (that introduce contact overlaps), the gap between the two policies becomes higher than ten percent. Finally, we observe that the LRU policy underperforms both policies, in sparse scenarios, while it converges to the (worse) *MP* policy in dense ones. This is reasonable, as LRU approximates an LFU policy (least frequently used), i.e., storing the most popular content when the popularity is stationary during the considered window. While in some scenarios LRU is actually not far from the results of our allocation strategy, it should be highlighted that we ignore the extra backhaul cost due to the frequent cache updates for LRU (that *optimal* does not have). In fact, simulations have shown that this "additional seeding" in LRU considerably degrades performance.

Buffer Capacity. In Figure 3b we vary the cache storage per vehicle between 0,02 and 0,5 percent of the catalogue (where the number of vehicles is 531 and the mean video length is one hour). Interestingly, the smallest storage capacity still achieves considerable performance gains. E.g., if one considers an entire Torrent or Netflix catalogue (~3 PB), a mobile helper capacity of about 500 GB (0, 02 percent) already suffices to offload 30 percent of the total traffic. Moreover, for small caches (less than 0, 05 percent of catalogue), *optimal* almost doubles the gain (from 18 to 30 percent) compared to the other policies. Finally, the random policy ignores the skewed Internet content popularity, and thus performs very poorly in all scenarios.

Video Length. Stationary regime analysis can be considered as a good approximation when there is a reasonable number of busy plus idle buffer playout periods, such that the transitory phase becomes negligible. In order to increase the number of these periods, the average content size needs to be large enough, given fixed mobility statistics. Figure 3c shows the fraction of data offloaded by the vehicular cloud for a set of content of the same length. As proved in Proposition 4.11, gains become larger according to the average video length. However, this increase is only marginal in the majority of the scenarios: in fact, even small content (15 minutes) provides a gain which is comparable to the asymptotic gain, validating the stationary regime analysis.

C. Mobile vs. Static Helpers

In this section, we verify the pertinence of the vehicular cloud, that is based on mobile helpers, against the femto-caching framework described in Golrezaei *et al.* [11], that is based on static SCs equipped with storage. In this second network, SC helpers are distributed in the considered area



(a) Offloading gain vs. number of vehicles. (b) Offloading gain vs. cache size. (c) Offloading gain vs. file size.

Fig. 3: % of data offloaded as a function of cache density (Figure 3a), buffer capacity (Figure 3b) and video length (Figure 3c).

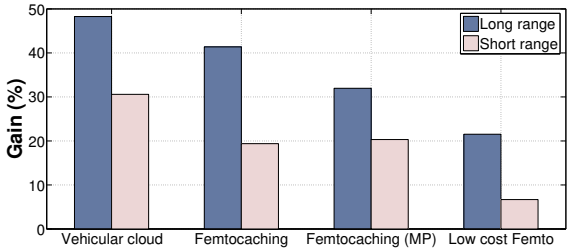


Fig. 4: Data offloaded for vehicular cloud and femtocaching.

proportionally to the popularity density, i.e., areas with a higher number of requests have higher SC density (this is a common operator policy since SCs are deployed to alleviate traffic "hotspots"). Users move according to the previously described SLAW trace, and they can also download video chunks at low cost from a nearby SC if it stores the requested video. Content is allocated using the algorithm described in Golrezaei *et al.* [11]. We consider two densities of SCs:

- *Femtocaching (equal number of helpers)*. From the analysis of the Cabspotting trace, the average number of vehicles *simultaneously* inside the area considered is lower than 200. In order to have a fair comparison with the vehicular cloud, we set the number of SCs to 200 where each SC has the same cache capacity as vehicles.
- *Low cost femto (equal cost)*. The capital expenditure of a SC consists of base station equipment, professional services (planning, installation and commissioning), and backhaul transmission equipment. This cost may range from 1000 € for a femtocell to 20.000-30.000 € for a microcell [30]. In the proposed vehicular cloud, the equipment might be pre-installed, and a large part of the OPEX could also be avoided as explained earlier. In fact, a first implementation of a similar vehicular cloud, where vehicles act as proxies, has shown a ten-fold cost reduction compared to SCs [1]. We therefore also consider a sparser deployment that equalizes the total cost where we set to 50 the number of SCs.

Figure 4 compares vehicular cloud and femtocaching in terms of data offloaded. We also simulate a femtocaching scenario with the *MP* policy. As expected, gains provided by the vehicular cloud are considerably higher than femtocaching for both short and (mainly) long range communications. This result is even more interesting considering the cost: in fact, storing content in vehicles permits almost 2,5 times higher gains than femtocaching with equal cost.

D. Per-Chunk Caching Strategy Evaluation

In this subsection, we evaluate the performance of the per-chunk policy described in Section V. We use that same real-trace based simulator introduced in Section VI-A, where a user abandons the playout of the video with some probability: when a user watches a chunk, she decides to watch the following chunk with probability $1-q$ or to abandon the playout with probability q (we set $q = 0,05$ as default value). Content is split in 10 chunks. Our main goal in this preliminary evaluation is to highlight the improvements, in terms of number of chunks offloaded, brought by caching per-chunk (when Ψ_{ij} is given by Eq. (11)) compared to per-content, in a realistic scenario.

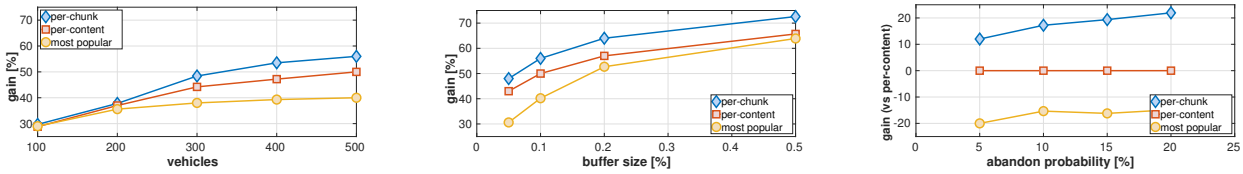
In Figure 5a we perform sensitivity analysis according to the number of vehicles h in the cloud which varies from 100 to 500. As the number of vehicles increases, the per-chunk policy offloads much more traffic compared to the other policies. For example, with 500 vehicles, this policy can offload almost 60 percent of the traffic. Figure 5b compares different buffer capacities per vehicle. Buffer size is in the range 0,05-0,5 percent of the catalogue (where $h = 531$). Considerable performance gains (i.e., more than 70 percent of traffic offloaded) can be achieved with very reasonable storage capacities (i.e., less than one percent of the catalogue). In Figure 5c we perform sensitivity analysis according to the probability of viewing the subsequent chunk. We analyse the range 5-20 percent (i.e., a video is entirely played out from 10 to 60 percent of times). In this scenario, the per-chunk policy provides a relative improvement between up to 25 percent compared to the per-content policy. What is more, the relative traffic offloaded by the per-chunk policy improves as the abandon probability increases which confirms that reshuffling chunks is an effective strategy to improve traffic offloading. Finally, we believe that a finer per-chunk policy (i.e., with a more realistic function for Ψ_{ij}) would provide larger gains.

VII. Discussion

A. Architecture

Hybrid LTE - D2D system. In the simplest case, vehicular helper nodes could act as end users (i.e., UEs) in an LTE system. In other words, the "backhaul" link ($\mathcal{I}-\mathcal{H}$) of our hybrid system is just a regular downlink between an eNodeB (the standard LTE BS) and a UE, over which content is pushed during off-peaks[11]. The "fronthaul" link ($\mathcal{H}-\mathcal{U}$) could then

[11]Note that content cannot usually be pushed at night to our mobile helpers. Nevertheless, we believe that within 24h there are enough traffic troughs to be able to update the caches from day to day, without congesting the network.



(a) Offloading gain vs. number of vehicles. (b) Offloading gain vs. cache size. (c) Offloading gain vs. abandon rate.

Fig. 5: Performance evaluation.

be operated as a D2D link. Previous work has confirmed the feasibility of opportunistic connections between vehicles and UEs [6]. IEEE 802.11p, which has been developed for the specific context of vehicular networks, is the de facto standard offering simplicity (uncoordinated access mechanism, no authentication) and low delay (few hundreds *ms* in crowded areas). An alternative to 802.11p is given by LTE Proximity Services (ProSe). LTE ProSe improves overall throughput, spectrum utilization and reliability at the cost of higher latency. However, Kim *et al.* [18] show that LTE ProSe is capable of satisfying delay requirements for most vehicular applications. What is more, LTE ProSe optimizes the battery usage since the user does not have to periodically probe nearby vehicles.

LTE-integrated system. A higher integration with cellular infrastructure could also be envisioned, where the $\mathcal{H}$ nodes are operated as (mobile) LTE relays [31] with local caches, and end users devices ($\mathcal{U}$) as regular UEs that can communicate with both macrocells and relays. The $\mathcal{I}-\mathcal{H}$ link is then a real backhaul link, while the $\mathcal{H}-\mathcal{U}$ a regular eNodeB-UE link.

B. Incentives

Modern cities might decide to offer Wi-Fi features in public transportation by installing inexpensive access points and storage into buses or trams to increase the number of passengers and generate additional revenues [1]. In this way, vehicular networks can produce real-time city-scale data from cheap sensors which can be used to increase safety and efficiency of municipal operations (e.g., traffic, waste collection). Additionally, MNOs could make deals with private bus or taxi fleets (e.g., Uber) providing the mobile relay and storage for free (thus improving, e.g., Uber customer experience). Regarding private transportation, it is estimated that by 2020 around 90 percent of all new cars are likely to have Internet connectivity. As a result, cellular operators see cars as another device to be connected to their networks, and they have started to propose data plan dedicated to vehicles (e.g., AT&T in US). MNOs might offer economic incentives (e.g., subscription reductions) to users that decide to join the vehicular cloud with their private vehicles. This increases their market share while offloading part of the mobile traffic.

C. Additional Use Cases

We believe one of the key strengths of our framework is its wider applicability. We present three additional use cases:

Femtocaching. In the femtocaching setup, while some of the mobility assumptions made do not exactly hold, as for example when the helpers are static and UEs are moving, our analytical formulas can still serve as a good approximation.

Secondary low cost operator. A new low cost operator could try to take up a market share by offering less expensive access to their (static or mobile) SCs to users with dual subscriptions. In this case, the optimization problem is rather for the low cost operator that aims to maximize the amount of traffic that it can serve from its own nodes.

Device-based caching. More futuristic scenarios that use device-based caching and device-to-device communication as the "secondary" inexpensive network of helpers [10] could also be tackled with our framework. For example, the helper $\mathcal{H}$ nodes could correspond to an initial set of UEs that the operator pushes content to. If these UEs can further distribute the content to other UEs (which then also become helpers) it becomes an interesting problem to transform the statistics of the size of this non-constant helper set, into the playout buffer idle statistics.

VIII. Conclusion

In this paper, we have focused our study on caching multimedia content. Video streaming has become dominant in the current Internet traffic [2]. We have exploited the fact that later chunks introduce an intrinsic delay tolerance on the content download. Using queueing theory notions, we were able to model the intermittent contacts with the mobile caches. We have provided the following main contributions: (i) we have modelled the playout buffer at the user device with a bulk queue where arrivals correspond to the bytes opportunistically downloaded from vehicles, and the service rate corresponds to the playout video rate. Moreover, we have added an additional queue to deal with overlapping meetings; (ii) we have calculated the number of bytes to download from the vehicular cloud based on the above queueing model. Then, we have formulated an optimization problem to infer the optimal per-content allocation and a subsequent per-chunk adaptation; (iii) finally, we have implemented a trace-driven simulator to validate the theoretical findings. We have also compared the vehicular cloud with the femtocaching framework [11], and performed cost analysis that has highlighted the advantages of the vehicular cloud as a function of cost reduction.

Acknowledgment

This work was funded by the French Government (National Research Agency, ANR) through the "Investments for the Future" Program reference #ANR-11-LABX-0031-01.

REFERENCES

[1] Veniam. https://veniam.com/.

[2] Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update. 2016-2021.

[3] H. Ahlehagh and S. Dey. Video-aware scheduling and caching in the radio access network. *IEEE/ACM Transactions on Networking*, 2014.

[4] N. Alliance. NGMN 5G White Paper. https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0.pdf, 2015.

[5] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting Mobile 3G Using WiFi. In *Proceedings of MobiSys*. ACM, 2010.

[6] V. Bychkovsky, B. Hull, A. Miu, H. Balakrishnan, and S. Madden. A measurement study of vehicular internet access using in situ wi-fi networks. In *Proc. of Mobile Computing and Networking*. ACM, 2006.

[7] N. Cheng, N. Lu, N. Zhang, X. S. Shen, and J. W. Mark. Vehicular WiFi Offloading. *Vehicular Communications*, Jan 2014.

[8] V. Conan, J. Leguay, and T. Friedman. Characterizing Pairwise Inter-contact Patterns in Delay Tolerant Networks. In *Proceedings of Autonomics*. ICST, 2007.

[9] S. K. Dandapat, S. Pradhan, N. Ganguly, and R. Roy Choudhury. Sprinkler: Distributed Content Storage for Just-in-time Streaming. In *Proceedings of CellNet*. ACM, 2013.

[10] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire. Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution. *IEEE Communications Magazine*, 2013.

[11] N. Golrezaei, K. Shanmugam, A. Dimakis, A. Molisch, and G. Caire. Femtocaching: Wireless video content delivery through distributed caching helpers. In *IEEE INFOCOM*, 2012.

[12] P. Hall. Heavy traffic approximations for busy period in an M/G/∞ queue. *Stochastic Processes and their Applications*, 1985.

[13] M. Harchol-Balter. *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge University Press, 2013.

[14] T. Hossfeld, S. Egger, R. Schatz, M. Fiedler, and C. Lorentzen. Initial delay vs. interruptions: Between the devil and the deep blue sea. In *Workshop on Quality of Multimedia Experience*, Jul 2012.

[15] T. Karagiannis, J.-Y. Le Boudec, and M. Vojnović. Power law and exponential decay of inter contact times between mobile devices. In *Proceedings of MobiCom*. ACM, 2007.

[16] S. Karlin and H. Taylor. *A First Course in Stochastic Processes*. Elsevier Science, 2012.

[17] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems.* Springer, 2004.

[18] H.-Y. Kim, D.-M. Kang, J.-H. Lee, and T.-M. Chung. A Performance Evaluation of Cellular Network Suitability for VANET. *J. of Electrical, Computer, Energetic, Electronic and Communication Engineering*, 2012.

[19] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer Publishing Company, Incorporated, 2007.

[20] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong. SLAW: A New Mobility Model for Human Walks. In *IEEE INFOCOM*, Apr 2009.

[21] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong. Mobile Data Offloading: How Much Can WiFi Deliver? *IEEE Transactions on Networking*, 2013.

[22] A. Mahmood, C. Casetti, C. F. Chiasserini, P. Giaccone, and J. Harri. Mobility-aware edge caching for connected cars. In *WONS*, 2016.

[23] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc., New York, NY, USA, 1990.

[24] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser. DAD data set epfl/mobility (v. 2009-02-24), 2009.

[25] K. Poularakis, G. Iosifidis, A. Argyriou, and L. Tassiulas. Video delivery over heterogeneous cellular networks: Optimizing cost and performance. In *IEEE INFOCOM*, Apr 2014.

[26] P. Raghavan and C. D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 1987.

[27] S. Ross. *Stochastic Processes*. Wiley series in mathematical statistics. Probability and mathematical statistics. Wiley, 1983.

[28] N. Sapountzis, S. Sarantidis, T. Spyropoulos, N. Nikaein, and U. Salim. Reducing the energy consumption of small cell networks subject to qoe constraints. In *IEEE Global Communications Conference*, Dec 2014.

[29] M. Sathiamoorthy, A. G. Dimakis, B. Krishnamachari, and F. Bai. Distributed storage codes reduce latency in vehicular networks. *IEEE Transactions on Mobile Computing*, Sep 2014.

[30] Senza Fili Consulting. The economics of small cells and wi-fi offload. 2013.

[31] S. Sesia, I. Toufik, and M. Baker. *LTE, The UMTS Long Term Evolution: From Theory to Practice*. Wiley Publishing, 2009.

[32] Small Cell Forum. Backhaul technologies for small cells: Use cases, requirements and solutions. 2013.

[33] G. Szabo and B. A. Huberman. Predicting the Popularity of Online Content. *Communications of the ACM*, Aug. 2010.

[34] L. Vigneri, T. Spyropoulos, and C. Barakat. Storage on wheels: Offloading popular contents through a vehicular cloud. In *IEEE WoWMoM*, Jun 2016.

[35] M. Zeni, D. Miorandi, and F. De Pellegrini. YOUStatAnalyzer: a tool for analysing the dynamics of YouTube content popularity. In *Proceedings of Valuetools*, 2013.

[36] F. Zhang, C. Xu, Y. Zhang, K. K. Ramakrishnan, S. Mukherjee, R. Yates, and T. Nguyen. EdgeBuffer: Caching and prefetching content at the edge in the MobilityFirst future Internet architecture. In *WoWMoM*, Jun 2015.

[37] Y. Zhang, J. Zhao, and G. Cao. Roadcast: A Popularity Aware Content Sharing Scheme in VANETs. In *IEEE Conf. on Distributed Computing Systems*, Jun 2009.

[38] J. Zhao and G. Cao. VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks. *IEEE Trans. on Vehicular Technology*, May 2008.

[39] X. Zhuo, W. Gao, G. Cao, and S. Hua. An incentive framework for cellular traffic offloading. *IEEE Trans. on Mobile Computing*, Mar 2014.

[40] X. Zhuo, Q. Li, W. Gao, G. Cao, and Y. Dai. Contact duration aware data replication in delay tolerant networks. In *IEEE Conference on Network Protocols*, Oct 2011.

Luigi Vigneri obtained his Master of Science in Computer Engineering from the Politecnico di Torino, Italy. During his studies, he also joined a double degree program and obtained a Master of Science in Computer Science from Telecom ParisTech, France. He is currently PhD candidate at EURECOM, Sophia Antipolis, France. His main research interests concern network modelling and optimization, and performance analysis.

Thrasyvoulos Spyropoulos received the Diploma in Electrical and Computer Engineering from the National Technical University of Athens, Greece, and a Ph.D degree in Electrical Engineering from the University of Southern California. He was a post-doctoral researcher at INRIA and then, a senior researcher with the Swiss Federal Institute of Technology (ETH) Zurich. He is currently an Assistant Professor at EURECOM, Sophia-Antipolis. He is the recipient of the best paper award in IEEE SECON 2008, and IEEE WoWMoM 2012, and runner-up for ACM Mobihoc 2011, and IEEE WoWMoM 2015.

Chadi Barakat is permanent researcher in the Diana group at INRIA, Sophia Antipolis. He got his master, PhD and HDR degrees in Computer Science from the University of Nice, Sophia Antipolis. He was with the LCA department at EPFL-Lausanne for a post-doctoral position, and a visiting faculty member at Intel Research Cambridge. Chadi Barakat was general chair for ACM CoNEXT 2012, PAM 2004 and WiOpt 2005 workshops, area editor for the ACM CCR journal and is currently on the editorial board of Elsevier Computer Networks. His main research interests are in Internet measurements and traffic analysis, user quality of experience, content-centric, software-defined and mobile wireless networking. Chadi Barakat is senior member of the IEEE and member of the ACM.