Storage on Wheels: Offloading Popular Contents Through a Vehicular Cloud

Luigi Vigneri, Thrasyvoulos Spyropoulos, Chadi Barakat

Storage on Wheels: Offloading Popular Contents Through a Vehicular Cloud

Luigi Vigneri
Institut EURECOM
Sophia Antipolis, France
Email: luigi.vigneri@eurecom.fr

Thrasyvoulos Spyropoulos
Institut EURECOM
Sophia Antipolis, France
Email: thrasyvoulos.spyropoulos@eurecom.fr

Chadi Barakat
INRIA
Sophia Antipolis, France
Email: chadi.barakat@inria.fr

***Abstract*—The increasing demand for mobile data is overloading the cellular infrastructure. Small cells and edge caching is being explored as an alternative, but installation and maintenance costs for sufficient coverage are significant. In this work, we perform a preliminary study of an alternative architecture based on two main ideas: (i) using vehicles as mobile caches that can be accessed by user devices; compared to small cells, vehicles are more widespread and require lower costs; (ii) combining the mobility of vehicles with delayed content access to increase the number of cache hits (and reduce the load on the infrastructure). Contrary to standard DTN-type approaches, in our system max delays are guaranteed to be kept to a few minutes (beyond this deadline, the content is fetched from the infrastructure). We first propose an analytical framework to compute the optimal number of content replicas that one should cache, in order to minimize the infrastructure load. We then investigate how to optimally refresh these caches to introduce new contents, as well as to react to the temporal variability in content popularity. Simulations suggest that our vehicular cloud considerably reduces the infrastructure load in urban settings, assuming modest penetration rates and tolerable content access delays.**

I. Introduction

The rapidly increasing demand for mobile data is menacing cellular operators. Upgrading the architecture to 4G is expensive, making operators reluctant. Densification through small cells is proving to be a bigger hassle and investment than initially expected [6]. What is more, introducing small cells requires significant upgrades to the backhaul network, which is predicted to become the new bottleneck [17].

Caching (popular) contents at the edge of the network (e.g., in small cells) has been proposed to alleviate the load on the backhaul and core network [9], [19], [31]. The utility of such caching systems is supported by recent studies on content popularity, demonstrating a significant overlap between user demands. Nevertheless, to ensure a high enough amount of requests are served by small cells, and thus significantly reduce the load on the main infrastructure, extensive coverage by small cells is necessary [32], which raises again concerns for CAPEX/OPEX costs.

To this end, in this paper we propose and study an alternative architecture, based on two main ideas. *First*, we propose to use vehicles as mobile small cells and data caches. These caches are controlled by the ISP (e.g., over a cellular interface [3]) and can be accessed directly by mobile devices (e.g. using WiFi or 802.11p [28]). In urban environments, the number of vehicles (private or public) is expected to be considerably higher than any envisioned small cell deployment. Furthermore, the related CAPEX/OPEX costs are expected to be lower, as many future vehicles will already be equipped with wireless communication equipment and storage [3], and some of the involved costs can be delegated, as in the femtocell model. While there exist other works that have considered content storage in vehicles [37], [38], this is the first to consider such vehicles as a common cloud maintained by an ISP.

Second, we propose to exploit the mobility of vehicles to serve more content requests, locally, from vehicular caches. Specifically, in our architecture, a user requests a content initially from the infrastructure (which redirects the user to the vehicular cloud) or queries nearby vehicles using WiFi. However, a key component in our system is that, if the content is not immediately available at a nearby vehicle, the user agrees to wait few minutes until any car with the content moves within range. Such delayed content access can increase the number of cache hits and reduce the load on the infrastructure, compared to the case of static, small cell caches.

Delayed content access has been widely investigated in the context of delay tolerant and opportunistic networking [10], [22] as well as for WiFi-based offloading [8], [27]. However, contrary to most such "DTN-type" approaches, in our system maximum delays are *guaranteed*, and kept to a *few minutes*: beyond a Time-To-Live (TTL) agreed between the ISP and the user, the content is fetched from the infrastructure. Such additional waiting delays could be easily amortized for large content transmissions (e.g., videos, software downloads), or be acceptable based on user subscription level (e.g. some users might be willing to pay cheaper plans and live with the *occasional* longer delays [20]) and context (e.g. roaming users might be more willing to wait for a low cost access).

While deadlines or TTLs are also considered in some WiFi-based offloading proposals, the sparseness of WiFi coverage implies that considerably larger deadlines, e.g. in the order of hours, are required to achieve significant offloading gains [25]. Furthermore, device-based offloading systems [34], while comprising a comparable number of mobile caches, face major resource constraints (battery lifetime, storage space, etc.) that vehicles do not face, raising significant concerns about the likelihood of such approaches being adopted in the near future.

The basic communication protocol for our "Storage on Wheels" system is defined by six steps (Fig. 1): (1) base stations or macro-cells push popular contents in vehicles; we refer to this action as *seeding*; (2) a user requests a content to nearby vehicle: (3) if the content is found (*cache hit*), the user can immediately download it; (4) otherwise he waits for new

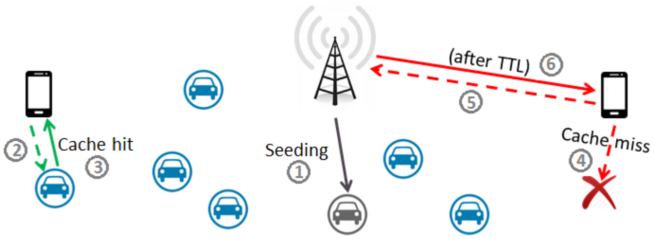


Fig. 1: Basic communication protocol of the offloading mechanism for the proposed infrastructure.

vehicles. (5) When TTL expires and the content has not been found in the vehicles (*cache miss*), the user contacts the cellular network (6) to get the content directly from the infrastructure. In this work, we study how to optimally allocate contents in order to minimize the load on the cellular infrastructure. Our main contributions in this paper are the following:

- We study analytically the problem of optimal content allocation, and derive the optimal number of copies of each content to be allocated to vehicles in the cloud, assuming a stable catalogue and average content popularity;
- We extend our analysis to more practical settings where connectivity can be lost while retrieving the content (due to mobility, interference, etc.) or (large) content is retrieved from multiple vehicles (e.g. in differently sized chunks);
- We study how to refresh the vehicles caches dynamically, to react to new content and changing content popularity;
- We use real traces for content popularity and mobility to study the feasibility of our system, and show that considerable offloading gains can be achieved even with modest technology penetration ($< 1\%$ of vehicles participating in the cloud) and reasonable *max* delays (1-5min).

Summarizing, this work is structured as follows: in Section II, we present the model and the general formulation of the content allocation problem; next, in Section III, we provide closed form expressions for the optimal allocation when vehicles caches are initially empty; then, we present an algorithm to periodically update the caches according to the variability of the content popularity in Section IV; after that, we validate our results through simulations in Section V; finally, we review related work in Section VI and conclude our paper with a summary and future work in Section VII.

II. Problem formulation

In this section, we first describe the proposed architecture and the system model. Then, we add some implementation details of the system. Finally, we define the cost function to minimize in order to optimally allocate contents in the vehicles. The main notation used in the paper is summarized in Table I.

TABLE I: Notation used in the paper.

α	Number of vehicles
β	Buffer size per vehicle
ν	Meeting rate
TTL	Time that the user is willing to wait before getting the content
$\varphi_t^{(i)}$	Popularity rate for content i at time t
$N^{(i)}$	Number of copies stored in the vehicles for content i
k	Number of contents in the catalogue
Δt	Refresh time and time window considered in the optimization
p	Probability to successfully download a content during a meeting

System model. We have a network with three types of nodes:

- *Infrastructure nodes* ($\mathcal{I}$), i.e. base stations or macro-cells. Their role is to serve users with the contents requested and to fill the vehicles buffers;
- *Cloud nodes* ($\mathcal{C}$), i.e. all the vehicles that can store contents (cars, buses, taxis, etc.), where $|\mathcal{C}| = \alpha$. Contents are pushed (*seeded*) to $\mathcal{C}$ nodes directly from $\mathcal{I}$ nodes. We assume that vehicles do not exchange contents with each other; inter-vehicular communication (V2V) is left as future work;
- *Mobile nodes* ($\mathcal{M}$), i.e. the users asking for online contents through their mobile devices, such as smartphones, tablets or netbooks. These nodes do not store any content, rather they ask for them to $\mathcal{C}$ or $\mathcal{I}$ nodes. Specifically, a communication between $\mathcal{C}$ and $\mathcal{M}$ can take place when their distance is less than a certain communication range (*contact*). Finally, we assume that the amount of time a user is willing to wait is smaller than the time window over which we are optimizing.[1]

Let K denote a set of contents available, where $|K| = k$. We refer to it as *catalogue*. Each content is assumed to have a known popularity rate $\varphi^{(i)}$ during a given time window Δt, calculated as $\varphi^{(i)} \triangleq 1/\Delta t \int_0^{\Delta t} \varphi_t^{(i)} \partial t$. To keep analysis tractable, we assume that all contents have equal size, e.g. equal to the mean file size in the catalogue. While this assumption is not true in reality, our model trivially allows to absorb the heterogeneity in the contents size: in fact, large contents can be split in chunks of equal smaller size, where chunks for the same content have the same popularity (and thus obtain the same number of replicas).

Finally, we assume that the mobility between $\mathcal{C}$ and $\mathcal{M}$ nodes leads to a contact process between them exhibiting exponential pairwise inter-contact times with rate ν. While inter-contact time distributions are still a subject of debate, several studies suggest that this is a reasonable assumption, especially in the tail of the distribution [12] [29] [24].

Architecture. ISP providers often offer low cost data plans with limited bandwidth, because of the increasing demand for data. However, the bandwidth offered is usually not enough to satisfy the users' needs. Thus, we propose our infrastructure as an additional feature to boost these cheap data plans. Basically, a user can browse the Internet contacting directly his cellular provider as usual. The cellular provider might decide to redirect the request to the vehicles for the popular contents as it happens in the CDN context, if the user has subscribed for the "Storage on Wheels" additional feature. Recall that the ISP is aware of what contents are popular or not. This setup might be really interesting for roaming users as well, that do not want to use expensive data plans abroad.

An alternative use case is to consider querying directly vehicles through WiFi. Because of the immense size and of the variety of the catalogue, we think that is more profitable

[1] In fact, in our system TTL $<< \Delta t$, since we only consider TTLs in the order of a few minutes. This is in contrast with some related work which use much longer TTLs to obtain similar gains [25].

in terms of cache hits to limit the user's choice to a smaller number of contents. For instance, the user can ask contents based on a generic list of popular contents updated by the $\mathcal{I}$ nodes. This list could be directly downloaded from the $\mathcal{I}$ or $\mathcal{C}$ nodes or in alternative ways (e.g., NFC tags or QR codes at bus stops). Finally, our architecture can naturally provide additional services such as traffic information, advertisement, events, news, P2P applications, etc. However, the definition of a complete and detailed architecture is beyond the scope of this paper as we focus on performance optimization.

Problem formulation. In our model, both pushing a copy in the vehicular cloud and having a cache miss *cost* a transmission. The final goal is to minimize the load on the infrastructure, i.e. the number of transmissions. To this end, we choose the cost function to be equal to the sum of the number of copies to be seeded ($\mathcal{I} \longrightarrow \mathcal{C}$) and the *cache misses* that occur in the considered time window Δt ($\mathcal{I} \longrightarrow \mathcal{M}$). Both of these require resources (e.g. bandwidth) from $\mathcal{I}$ to be expended, and thus should be minimized. Yet, reducing the one often leads to increasing the other. Hence, we can write the corresponding optimization problem in order to minimize the number of accesses to the $\mathcal{I}$ nodes:

$$\begin{aligned} \min_{N^{(i)}} \quad & \sum_{i=1}^{k} \left(\Delta t \varphi^{(i)} \mathbf{P}[X^{(i)} > TTL] + \max\left(N^{(i)} - N_0^{(i)}, 0 \right) \right) \\ \text{s.t.:} \quad & 0 \leq N^{(i)} \leq \alpha, \quad i = 1, \ldots, \text{k}, \\ & \sum_{i=1}^{k} N^{(i)} \leq \alpha\beta, \end{aligned} \tag{1}$$

where $N^{(i)}$ is the number of copies stored in the vehicles for the content i, $N_0^{(i)}$ is the *initial* number of copies and $X^{(i)}$ is a random variable corresponding to the time needed to successfully download content i.

The first term of the objective function takes into account the number of *cache misses* occurring in Δt: $\Delta t \varphi^{(i)}$ is the expected number of *requests* for content i during Δt. Every time that a requested content is not provided by the vehicles within TTL, we have a *cache miss* that increases the cost. $\mathbf{P}[X^{(i)} > TTL]$ is the probability not to download the whole content within TTL. Hence, the product of these three terms is equal to the total number of missed requests in the time window. The second term corresponds to the *incremental seeding cost* for the communication $\mathcal{I} \longrightarrow \mathcal{C}$, i.e. the cost of adding *new* copies when buffers are not empty (we assume that removing a copy does not have a cost). The objective function is subject to the constraints:

- the number of vehicles having the content i cannot be negative: $N^{(i)} \geq 0$,
- the number of vehicles having the content i cannot be higher than the cardinality of $\mathcal{C}$ ($|\mathcal{C}| = \alpha$): $N^{(i)} \leq \alpha$,
- each vehicle has storage constraints and cannot store more than β contents[2]: $\sum_{i=1}^{k} N^{(i)} \leq \alpha\beta$.

In the following section, we discuss the particular case when the vehicles caches are empty ($N_0^{(i)} = 0$); then, in Section IV, we solve the optimization problem in Eq. (1) considering also the incremental seeding.

III. Optimal content allocation for large Δt

In this section, we consider how to optimally allocate contents in the vehicular cloud, when the caches are initially empty. This scenario can be considered as a *bootstrap phase* for the infrastructure, where the buffers need to be filled. Furthermore, when the operator defines a *large* Δt, we assume that the new allocation does not depend on the initial set of contents stored in the vehicles (i.e. $N_0^{(i)} = 0, \forall i \in K$), since content popularity substantially changes in large intervals. Thus, the operator needs to reboot all its caches. This scenario provides useful insights and methodology for subsequent results.

A. Baseline scenario

We first consider the baseline scenario, where we assume that if a $\mathcal{M}$ node encounters a $\mathcal{C}$ node, it will be able to receive the whole requested content with probability 1. Thus, we can derive the following lemma:

Lemma 1. *In the baseline scenario, the probability* not *to download the whole content within TTL is* $\mathbf{P}[X^{(i)} > TTL] = e^{-\nu TTL \cdot N^{(i)}}$.

Proof: The inter-meeting time between two contacts is assumed to be exponentially distributed. In this scenario, $X^{(i)}$ corresponds to the time needed to meet the first vehicle having the requested content, therefore $X^{(i)} \sim \exp(\nu \cdot N^{(i)})$. The result in Lemma 1 directly follows from elementary properties of the exponential distribution [16]. ■

By using Lemma 1, we obtain the following result:

Result 1. *Consider the problem described by Eq. (1). The optimal number of copies to allocate is given by:*

$$N^{(i)} = \begin{cases} 0, & \text{if } \varphi^{(i)} < L \\ \frac{1}{\nu TTL} \ln\left(\frac{\nu TTL \varphi^{(i)} \Delta t}{\gamma^{(i)}} \right), & \text{if } L \leq \varphi^{(i)} \leq U \\ \alpha, & \text{if } \varphi^{(i)} > U \end{cases}$$

where $L \triangleq \frac{1+\rho}{\nu TTL \Delta t}$, $U \triangleq \frac{(1+\rho)e^{\alpha\nu TTL}}{\nu TTL \Delta t}$, $\gamma^{(i)} \triangleq 1 - \mu^{(i)} + \lambda^{(i)} + \rho$ *and* $\mu^{(i)}$, $\lambda^{(i)}$ *and* ρ *are appropriate Lagrangian multipliers (non-negative since the constraints are formulated as inequalities).*

Proof: The problem in Eq. (1) is *convex* since the objective is the sum of convex functions and the constraints are all linear. Thus, it can be solved with the method of Lagrangian multipliers [7]. We convert Eq. (1) to the standard form (from minimization to maximization) and we write the Lagrangian function of the problem:

$$\begin{aligned} L = -\Delta t \sum_{i=1}^{k} \varphi^{(i)} e^{-\nu TTL \cdot N^{(i)}} - \sum_{i=1}^{k} N^{(i)} + \sum_{i=1}^{k} \mu^{(i)} N^{(i)} + \\ \sum_{i=1}^{k} \lambda^{(i)} \left(\alpha - N^{(i)} \right) + \rho \left(\alpha\beta - \sum_{i=1}^{k} N^{(i)} \right), \end{aligned} \tag{2}$$

where $\lambda^{(i)}$, $\mu^{(i)}$ and ρ are the Lagrangian multipliers. The corresponding *KKT* conditions are:

$$\begin{cases} \mu^{(i)} N^{(i)} = 0 \\ \lambda^{(i)} (\alpha - N^{(i)}) = 0 \\ \rho(\alpha\beta - \sum_{i=1}^{k} N^{(i)}) = 0 \end{cases} \tag{3}$$

[2] We do not need to put extra constraints for individual car capacities (i.e. no car having more than β contents), because we assume that all the vehicles are statistically identical.

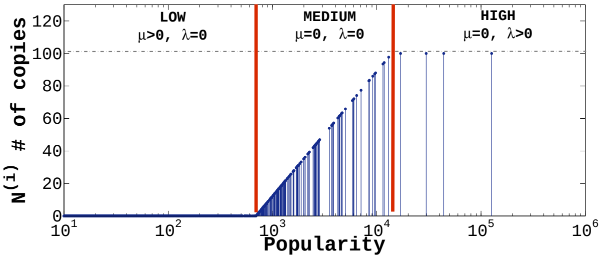


Fig. 2: Optimal allocation in semi-log scale ($\alpha = 100$).

The differentiation of Eq. (2) with respect to $N^{(i)}$ gives:

$$\frac{\partial L}{\partial N^{(i)}} = \nu TTL\varphi^{(i)}\Delta t e^{-\nu N^{(i)} TTL} - 1 + \mu^{(i)} - \lambda^{(i)} - \rho = 0$$

$$\Rightarrow N^{(i)} = \frac{1}{\nu TTL} \ln\left(\frac{\nu TTL\varphi^{(i)}\Delta t}{\gamma^{(i)}}\right). \tag{4}$$

Eq. (4) is subject to the constraints described by Eq. (3). Specifically:

- *Low popularity contents ($\mu > 0$)*: since the allocation cannot be negative, when the popularity is too low the number of copies to allocate is 0 (left side of Fig. 2). We are in this case when:

$$\frac{1}{\nu TTL} \ln\left(\frac{\nu TTL\varphi^{(i)}\Delta t}{\gamma^{(i)}}\right) < 0 \Rightarrow \varphi^{(i)} < \frac{1+\rho}{\nu TTL\Delta t} \triangleq L \tag{5}$$

  and

$$\mu^{(i)} = 1 + \rho - \nu TTL\varphi^{(i)}\Delta t. \tag{6}$$

- *High popularity contents ($\lambda > 0$)*: on the other hand, the number of copies in the cars is limited, upper-bounded by the number of cars available ($N^{(i)} = \alpha$) as we can see from the right side of Fig. 2. We are in this case when:

$$\varphi^{(i)} > \frac{(1+\rho)e^{\alpha\nu TTL}}{\nu TLL\Delta t} \triangleq U \tag{7}$$

  and

$$\lambda^{(i)} = \nu TTL\varphi^{(i)}\Delta t \cdot e^{-\alpha\nu TTL} - 1 - \rho. \tag{8}$$

As we can see from Eqs. (6) and (8), $\mu^{(i)}$ and $\lambda^{(i)}$ directly depend on the value of ρ. For this reason, a last step in deriving the exact allocation is just to infer the value of the Lagrangian multiplier ρ. Unluckily, since the value of ρ depends on the popularity of the entire set of contents K, it is not possible to determine a closed form expression for it. However, it can be proved that the cost function is *monotonically increasing* according to ρ (the proof is omitted due to space limitations and can be found in [1]), i.e. the cost is minimum for the minimum value of ρ that satisfies the capacity constraint. Moreover, if the capacity constraint is satisfied, ρ is trivially equal to 0. Thanks to these considerations, we can find this value of ρ through a simple minimum search algorithm. ■

According to the value of ρ, we now elaborate on the different solution regimes for the Medium popularity contents:

All constraints inactive ($\gamma = 1$): when all the constraints are inactive, the optimal allocation depends on the logarithm of the popularity (Fig. 2). Since here the buffer space is not an issue, the content allocation is decoupled, i.e. the optimal number of copies per content is independent of the popularity of the other contents. Finally, from Eq. (4), we can note that the optimal number of copies is attenuated by νTTL.

Capacity constraint active ($\gamma = 1 + \rho$): increasing the popularity or reducing the buffer space can violate the storage constraint. Even in this case, the allocation is still logarithmic on popularity. When the capacity is limited, the multiplier ρ reduces the optimal allocation as the number of contents increases or the total capacity decreases by increasing the thresholds L and U.

B. Connectivity loss scenario

When an $\mathcal{M}$ node is inside the communication range of a $\mathcal{C}$ node, the connectivity might be lost. This can be due to many reasons, such as network failures, vehicle mobility and interferences. In detail, a success or a failure in downloading a content depends on few key factors: (i) the contact duration, that is the amount of time during which $\mathcal{M}$ and $\mathcal{C}$ nodes can exchange data inside the communication range; (ii) the throughput that depends on the distance between the nodes: in fact, WiFi protocols are defined to use dynamic rate scaling, and the throughput will automatically decrease as the signal strength decreases, i.e. as the distance between the nodes increases; (iii) the size of the content; (iv) interferences and variability in the urban radio environment [26]. In this subsection, we propose two policies to face the problem of connectivity loss:

1) Repeat policy: In reality, it might be difficult to give an accurate estimation of the success/failure probability per contact. For this reason, we define the average probability p to successfully download a content during a meeting. In the repeat policy, the content download will restart from the beginning when an $\mathcal{M}$ node loses the connection with a vehicle (e.g., TCP session expires) and meets a new one.

Result 2. *The optimal content allocation for the repeat policy is given by:*

$$N^{(i)} = \begin{cases} 0, & \text{if } \varphi^{(i)} < L' \\ \frac{1}{p\nu TTL} \ln\left(\frac{p\nu TTL\varphi^{(i)}\Delta t}{\gamma^{(i)}}\right), & \text{if } L' \leq \varphi^{(i)} \leq U' \\ \alpha, & \text{if } \varphi^{(i)} > U' \end{cases}$$

where $L' \triangleq \frac{1+\rho}{p\nu TTL\Delta t}$ *and* $U' \triangleq \frac{(1+\rho)e^{p\alpha\nu TTL}}{p\nu TTL\Delta t}$.

Proof: Given the Poisson process with rate ν counting the contacts between $\mathcal{C}$ and $\mathcal{I}$ nodes, suppose that each event corresponds to a successful download with probability p. Then successful downloads form a Poisson process with rate $p\nu$. For this reason, we have:

$$\mathbf{P}[X^{(i)} > TTL] = e^{-p\nu TTL\cdot N^{(i)}}.$$

The rest of the proof proceeds as in the proof of Result 1. ■

2) Resume policy: Due to the limited contact duration and to the large size of the contents, downloading a content in one shot might be hard, leading to a small value of p. Alternatively, if we consider large contents, like videos, chunking is a popular way to break down the file into smaller pieces. Hence, during a contact a node could download one or more chunks. Moreover, new technologies allow easily to stop and resume the download at any time (e.g., latest versions of browsers, online music players, etc.). In the resume policy, when a $\mathcal{M}$ node loses the connection, the download will resume from the point of interruption during the following *cache hit*. Specifically, we suppose that for each meeting j, we can download r_j bytes of the content, where r_j is a continuous random variable with values $\in [0, +\infty)$. Furthermore, we define s_0 as the content size in bytes (recall that s_0 is assumed equal for any content).

Lemma 2. *Let $Y^{(i)} \triangleq \sum_{j=1}^{M^{(i)}} r_j$ be the number of bytes downloaded within TTL for content i, where $M^{(i)}$ indicates the number of cache hits in that interval. In the resume scenario, the probability* not *to download the whole content within TTL is equal to:*

$$\mathbf{P}[X^{(i)} > TTL] \equiv \mathbf{P}[Y^{(i)} < s_0] = \mathcal{L}^{-1}\left\{ \frac{e^{[R^*(s)-1]\nu TTL\cdot N^{(i)}}}{s} \right\}(s_0). \tag{9}$$

where $\mathcal{L}^{-1}\{F(s)\}(t)$ is the inverse Laplace transform of $F(s)$.

Proof: Since the inter-meeting time between two contacts is assumed to be exponentially distributed, $M^{(i)}$ follows a Poisson distribution. Furthermore, a *random* sum of identically distributed random variables has a Laplace transform that is related to the transform of the summed random variable and of the number of terms in the sum:

$$Y^{*(i)}(s) = M^{*(i)}(R^*(s)),$$

where $Y^{*(i)}$ (resp. R^*) is the Laplace transform of $Y^{(i)}$ (resp. r_j) and $M^{*(i)}$ is the Z-transform of $M^{(i)}$. The number of meetings within TTL is Poisson distributed, therefore we can write $Y^{*(i)}(s)$ as follows:

$$Y^{*(i)}(s) = e^{[R^*(s)-1]\nu TTL\cdot N^{(i)}}.$$

In the resume policy, the probability not to download the whole content within TTL is equal to the probability that the sum of the bytes downloaded within TTL ($Y^{(i)}$) is less than s_0. Since the cdf of f_X is given by $F_X(x) = \mathcal{L}^{-1}\left\{\frac{\mathcal{L}\{f_X\}}{s}\right\}(s_0)$, we can compute $\mathbf{P}[X^{(i)} > TTL]$ as in Eq. (9). ■

While the above formula is generic, it requires knowledge of the distribution of r, the "effective capacity" of contacts, which might be difficult to obtain. Furthermore, calculating the inverse transform is not trivial, except for very specific r distributions. To proceed and obtain a closed form estimate of this probability, observe that $Y^{(i)}$ is a compound Poisson process, because the number of meetings are Poisson distributed and the reward (bytes downloaded) in each contact is independent of the inter-contact times. According to this definition, we can derive the following result:

Result 3. *For large values of $M^{(i)}$, the optimal content allocation corresponding to the resume policy can be approximated using the solution of the following equation:*

$$N^{(i)} = \omega^{(i)} e^{-\omega^{(i)2}} \cdot \varphi^{(i)}\Delta t/(\gamma^{(i)}\sqrt{8\pi}),$$

where $\omega^{(i)} \triangleq (s_0 - \mathbf{E}[Y^{(i)}])/\sqrt{\text{Var}(Y^{(i)})}$.

Proof: Let $\mathbf{E}[Y^{(i)}] = \bar{r}\nu TTL \cdot N^{(i)}$ and $\text{Var}(Y^{(i)}) = (\bar{r}^2 + \sigma^2)\nu TTL \cdot N^{(i)}$ be the mean and variance of the compound Poisson process $Y^{(i)}$, where $\bar{r} \triangleq \mathbf{E}[r_j]$ and $\sigma^2 \triangleq \text{Var}(r_j)$ are respectively the mean and the variance of r_j. Especially in urban environments, the number of contacts is expected to be considerably large. When $M^{(i)}$ corresponds to large values, we could use a normal approximation for $Y^{(i)}$ using only the first two moments [33], i.e. $Y^{(i)} \sim \mathcal{N}(\mathbf{E}[Y^{(i)}], \text{Var}(Y^{(i)}))$; in this case, the probability *not* to download the whole content within TTL is:

$$\mathbf{P}[Y^{(i)} < s_0] \approx \Phi\left(\frac{s_0 - \bar{r}\nu TTL \cdot N^{(i)}}{\sqrt{(\bar{r}^2 + \sigma^2)\nu TTL \cdot N^{(i)}}}\right) \tag{10}$$

Finally, from Lemma 2 we know that $\mathbf{P}[X^{(i)} > TTL] \equiv \mathbf{P}[Y^{(i)} < s_0]$. Thus, we can replace Eq. (10) in Eq. (1) and solve with the method of the Lagrangian multipliers as in the proof of Result 1. ■

For this approximation to hold, the "stopping" $M^{(i)}$, i.e. the number of contacts until the whole content is retrieved, needs to be sufficiently large. However, if $Y^{(i)}$ is highly skewed, then this approximation might also fail. In this case, it is possible to use other approximations (e.g., gamma, Edgeworth).

IV. Dynamic adaptation to changing popularity

In this section, we consider a more practical setup, where caches are updated dynamically, as new contents are introduced, and/or existing contents exhibit a significant change in popularity. Adapting to changing content popularity is not only important to introduce new contents and delete obsolete ones, but also to increase the potential performance gains.

Specifically, suppose that video A and video B have the same number of views per day. Moreover, suppose that video A is very popular during the day, while video B is more popular during the night. We would like to capture this behaviour by allocating more copies for A during the day and for B during the night. Since they have the same average popularity over one day, if Δt is too large (e.g. one day) the two videos will be allocated the same number of copies. On the other hand, for a small Δt, the cache hits due to the additional copies allocated during the interval, are not large enough to amortize the cost of seeding these new copies, being perhaps removed before a newer allocation is selected in the next time window.

In such a system, seeding is *incremental*, taking into account the existing allocation, and adjusting it where necessary, depending on the potential gains predicted for a shorter time window (i.e., until the next update).

Result 4. *Consider that the initial allocation $N_0^{(i)}$ is given and not null. Then, given a Δt, the optimal number of copies to add (or remove) is:*

$$\Delta N^{(i)} = \frac{1}{\nu TTL}\ln\left(\frac{\nu TTL\varphi^{(i)}\Delta t}{\bar{\gamma}^{(i)}}\right) - N_0^{(i)}, \tag{11}$$

where $\Delta N^{(i)} \triangleq N^{(i)} - N_0^{(i)}$, $\bar{\gamma}^{(i)} \triangleq \mathbf{1}_A(\Delta N^{(i)}) - \mu^{(i)} + \lambda^{(i)} + \rho$ and $A = (0, +\infty)$.

Proof: The proof proceeds as in the proof of Result 1, by using the method of the Lagrangian multipliers. However, the derivative of the max function in Eq. (1) is equal to:

$$\frac{\partial \max(\Delta N^{(i)}, 0)}{\partial \Delta N^{(i)}} = \begin{cases} 0, & \text{if } \Delta N^{(i)} \le 0 \\ 1, & \text{if } \Delta N^{(i)} > 0 \end{cases}$$

i.e., the solution is different depending whether we add or remove copies. ■

As we can see from Result 4, the incremental seeding makes the problem of updating the caches non trivial, because $\Delta N^{(i)}$ appears also in $\bar{\gamma}^{(i)}$. Thus, we now propose a simple algorithm computing the number of copies to add or remove every Δt for each content. This heuristic allows to have an easy implementation in practice. Specifically, every time window, we make the decision if it is convenient adding more copies for a given content. Basically, seeding one more copy provides a gain equal to the number of cache misses saved by the additional copy. The gain is given by:

$$\text{gain}^{(i)} \triangleq \varphi^{(i)} \Delta t \cdot e^{-\nu TTL \cdot N^{(i)}} (1 - e^{-\nu TTL}).$$

Then, we sort the contents according to the gain that can provide and, if this is higher than the seeding cost, we add a copy (or more copies) in the cloud until the buffer is full or there no other contents to seed. On the other hand, if all the caches are full, storing new copies must follow the deletion of the less popular ones; removing one copy leads to a loss equal to the additional cache misses:

$$\text{loss}^{(i)} \triangleq \varphi^{(i)} \Delta t \cdot e^{-\nu TTL \cdot N^{(i)}} (e^{\nu TTL} - 1).$$

Then, we select the content with the highest gain and the one with the lowest loss. If $\max(\text{gain}) - \min(\text{loss}) > 1$, then the switching is advantageous. We call *switching* the action taken by the $\mathcal{I}$ nodes to remove a content and replace it with another one. We recompute the gain and the loss for the contents switched and we iterate until the condition is satisfied, i.e. there is at least one advantageous switching. We add/switch the contents every time window.

V. Simulation Results

We validate our model through MATLAB simulations. We simulate the load on the infrastructure and we compare it with other caching systems. In Section V-A, we consider the optimal allocation policies described in Section III and we analyse the impact of different parameters in the proposed cache system. Then, in Section V-B we test the performance of the contents replacement algorithm seen in Section IV. Finally, in Section V-C we perform simulations based on a real mobility trace to inspect the cost savings in a real world scenario.

In our analysis we consider the content popularity of YouTube videos, since a large percentage of mobile data traffic is represented by video files. We download from [4] a database generated with YouStatAnalyzer [36] that collects statistics for 100.000 YouTube videos. The database includes static (title, description, author, duration, related videos, etc.) and dynamic information (daily and cumulative views, shares, comments, etc.). In our simulations, we only take into account the number of views related to the last 360 days. However, these values are equal to the total number of views per day in the world, then we scale them properly[3]. We have also created synthetic traces based on the work in [13]. Simulations based on these synthetic trace confirm the observations made using the real trace. We therefore focus on the former.

Furthermore, we consider a square area 5000m x 5000m in the center of San Francisco. We use the *Cabspotting*[4] trace to compute the average meeting rate ν: we randomly place the $\mathcal{M}$ nodes and, considering a communication range of 200m, we calculate the meeting rate with each $\mathcal{C}$ node. We find $\nu = 4$ contacts/day. According to the density of the city and to the number of vehicles per capita, we estimate to 100.000 the number of vehicles in the area considered. However, in order to be realistic about initial technology penetration we assume that only 1% of these vehicles is participating in the cloud. We assume that each car can store 100 contents ($0, 1\%$ of the catalogue). We set TTL to 3 minutes. Unless otherwise stated, we will use these parameters summarized in Table II.

DESCRIPTION	PARAM	VALUE
Number of vehicles	α	1000 cars
Buffer size	β	100 contents/car
Meeting rate	ν	4 contacts/day
Time-To-Live	TTL	3 minutes
Number of contents	K	100.000 contents
Communication range		200m

TABLE II: Parameters of the scenarios considered.

Finally, we compare our allocation policies with:

- *No cache:* no contents are stored in the vehicles; the probability of miss is equal to 1, therefore the cost corresponds to the total demand: cost $= \Delta t \sum_{i=1}^{K} \varphi^{(i)}$;
- *Random:* contents are allocated randomly in the vehicles;
- *Square root:* this policy behaves similarly to our optimal allocation, but it replaces the logarithm with the square root, after an appropriate normalization to satisfy the storage constraint.

A. Optimal Content Allocation for Large Time Windows

We perform numerical simulations considering constant content popularity during a time window Δt, which is set to 1 week. Specifically, we compare the effect of buffer size, TTL and other parameters on the final gain comparing different policies. We show that the allocation policies presented in Section III clearly decrease the load on the infrastructure.

Fig. 3 depicts the cost in terms of total accesses during the period Δt, broken down into seeding cost and cache misses, for the various policies. Our policy reduces the total cost by around 65%, more than any other policy. What is more, it improves twice the performance compared to the *square root* policy, which is known to achieve optimal results in conventional peer-to-peer networks [11].

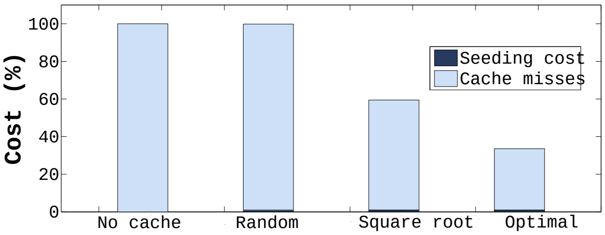


Fig. 3: Cost savings with different policies (Δt = 1 week) with numerical simulations.

In Fig. 4, we plot how TTL, buffer size and number of cars affect the final cost. Specifically, Fig. 4a shows the cost according to the value of TTL for the different policies. It is very important to note that considerable gains can be achieved with very small TTL values (in the order of a few minutes) and small number of vehicles participating in the cloud (1%).

[3] We scale them linearly taking into account the number YouTube users and the population of San Francisco

[4] GPS coordinates from 536 taxis in San Francisco over 23 days [30].

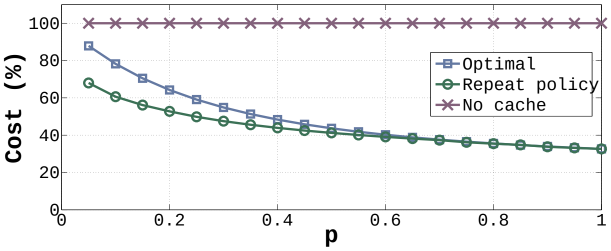


Fig. 5: Cost savings with different values of p.

This provides some evidence on the advantages of offloading based on a vehicular cloud, compared to offloading using small cells or WiFi, as for example in [8] or [25]. E.g, [8] reports minor gains for similar small deadlines, while [25] requires a much longer TTL (order of 1-2 hours) to achieve similar gains. In addition, increasing the TTL further has diminishing returns. This implies that even users not willing to wait too long could participate in such a system (benefiting themselves and the operator).

An efficient cache system should store as many popular contents as possible. However, in reality the catalogue of online contents is really large and only a small percentage of them can be stored. Fig. 4b shows the cost according to the buffer size. From the plot we can observe that storing 100 contents/car (only 0,1% of the total number of contents) provides a gain of almost 60%. In a scenario with a larger catalogue (e.g. 100 millions), it seems doable to store 0,1% of the contents (e.g. 100.000 contents/car) needed to achieve good savings. What is more, due to the intrinsic characteristics of the popularity distribution, the system might require an even smaller number of storage in order to achieve similar gains.

In an urban environment, the great availability of vehicles leads to large gains for the proposed infrastructure. However, an operator will probably keep using our framework even if the number of vehicles available decreases: in Fig. 4c we depict the cost savings according to the number of $\mathcal{C}$ nodes participating in the cloud and the gain observed is not less than 50% when more than 250 vehicles are part of the cloud.

So far we have assumed that the probability of downloading a content during a meeting is equal to 1. However, because of external factors, a user might not be able to get the requested content during a meeting. According to the repeat policy discussed in Section III-B1, in Fig. 5 we plot the percentage of savings for different values of p. We can note that, even when $p = 0.5$, i.e. a $\mathcal{M}$ node loses the connection during half of the downloads, we can have a gain of almost 60% in terms of total number of accesses to the core infrastructure. Clearly, this will be at the expense of some larger delay compared to the case of no disconnections. Furthermore, we plot the gain provided by the original allocation policy (calculated with $p = 1$) in a scenario with losses: the plot shows that it is important trying to estimate the value of p and tune the allocation accordingly, since this can bring up to the 20% of additional savings.

Finally, it has been long observed in many contexts, including Internet contents, that popularity exhibits strong skewedness. To evaluate the effect of such popularity differences, in Fig. 6 we do not take into account the real dataset, rather we consider bounded Pareto distributions (minimum value = 1 request/day, maximum value = 100.000 requests/day and ζ as shape parameter). We can observe that when the variance increases (ζ low), the optimal allocation brings a considerable gain up to the 70% in terms of number of accesses to the backbone. This is due to the fact that, if ζ is low, some contents have very high popularity, and caching them leads to a large number of cache hits. On the other hand, the gain goes to 0 when ζ increases, i.e. the differences in the content popularity are negligible making it hard to create enough cache hits with any subset of them that can fit in the cloud.

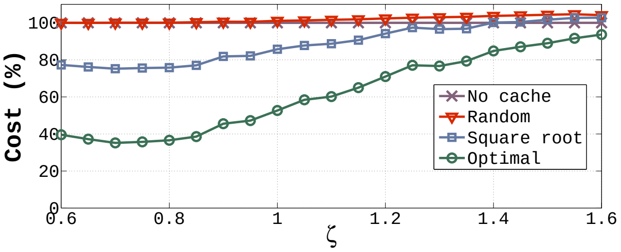


Fig. 6: Sensitivity analysis vs. popularity distributions.

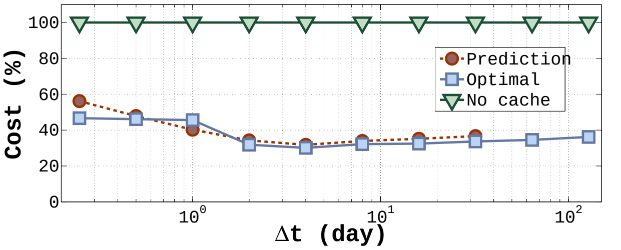
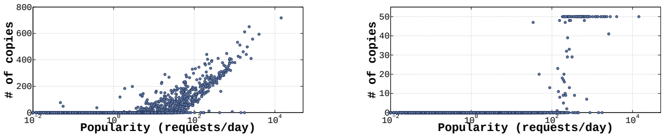


Fig. 7: Sensitivity analysis vs. Δt.

B. Dynamic Adaptation to Changing Content Popularity

In this subsection, we analyse how refreshing the caches affects the performance of the vehicular cloud. We still use the parameters described in Table II. Moreover, here we increase the realism of the simulations by considering that the popularity of the contents is not known in advance, but it requires to be estimated. Several studies have been done on content popularity prediction [18] [15]. In our simulator, we build a simple predictor to estimate the future popularity according to the previous samples, by using an Exponential Weighted Moving Average based on the latest 10 time windows. Building the best predictor goes beyond the scope of this paper. Here, we just want to show how an error in the prediction affects the system and if the considerations done are still valid.

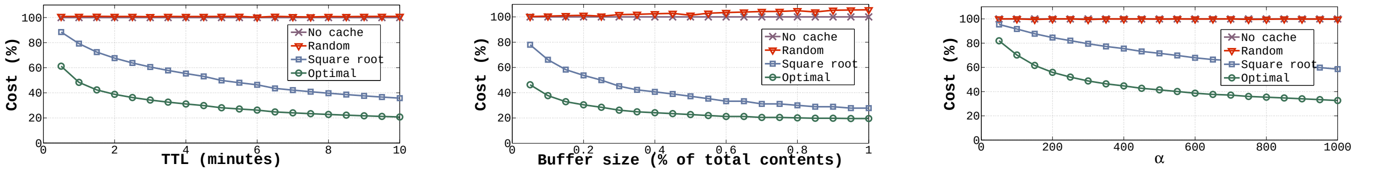
The content popularity provided by the database used [4] has daily granularity; however, several studies have shown a clear sinusoidal behaviour on a daily basis [18] [5]. We



(a) $\alpha = 1000$.

(b) $\alpha = 20$.

Fig. 8: Contents allocation ($\Delta t = 2$ days).



(a) Cost comparison according to TTL.

(b) Cost comparison according to buffer capacity.

(c) Cost comparison according to number of vehicles.

Fig. 4: Sensitivity analysis vs. TTL, buffer size and number of vehicles.

exploit these studies to estimate the content popularity on a hourly basis. Fig. 7 shows the final cost when caches are updated with a varying refresh time over a long time period to ensure capturing the variations in the content popularity (in the simulator, we set this long time period to 1 year). In this scenario, the vehicular cloud still provides a considerable number of savings (from 50% to 70% depending on Δt) in terms of accesses to the infrastructure. Moreover, we can have gains similar to the case with perfect knowledge (line "optimal") even by using a simple predictor.

In Fig. 8 we can observe a snapshot of the number of copies allocated per content according to its content popularity. We capture the snapshot at a random day of our trace. The plot is in semilog scale and $\Delta t = 2$ days. In this case, even if the shape globally follows the expected logarithmic behaviour, we can see a significant number of outliers. This is due to the fact that the number of copies does not change instantaneously when the popularity varies, due to the seeding cost. It is interesting to note that, in Fig. 8b, since the number of cars is limited, we observe a *single-threshold behaviour*, where the system will end up either caching contents everywhere (case of popular contents) or not caching them at all (case of non popular ones). Indeed, from Eqs. (5) and (7), when vehicles buffers cannot satisfy the large user demand, L and U get closer.

C. Validation through real mobility traces

In this subsection, we present the results of MATLAB simulations based on real mobility traces. We use the Cabspotting database [30], where Δt is equal to 23 days and the number of vehicles α is 536. In these simulations, we first generate a set of requests per content according to the popularity obtained by the YouTube database [4]. Then, we associate a timestamp and a location to each request. We assume that the number of requests in an area is proportional to the density of vehicles in that area and that the requests are concentrated during daytime. Once a request appears, we check if one of the vehicles in the communication range of the request within TTL has the required content. In this scenario, we set TTL to 5 minutes. However, despite we consider small buffer capacity per vehicle (0,1% of the total number of contents), we will show that, on average, contents are delivered much earlier than TTL. Furthermore, we assume that, when vehicles are closer to the location of the request, the contact duration and the throughput are larger, increasing the probability p to successfully deliver the content. For this reason, we consider p as inversely proportional to this distance, with mean equal to 0,7. We perform trace-based simulations comparing (i) repeat policy (called "optimal" in the figure), (ii) dynamic adaptation to the changing popularity ("optimal(var)"), where caches are updated on a daily basis, and non optimal policies.

Fig. 9 depicts the cost in terms of number of accesses to the infrastructure during the period Δt when the Cabspotting trace for mobility is used. The plot shows that repeat policy can decrease the load on the infrastructure until 40% even if only few hundreds of vehicles (with small buffer capacity) are participating into the cloud. Similar cost savings are provided when we allow to vary the content allocation on daily basis. What is more, users usually wait much less than TTL in order to download the requested content: simulations revealed that, in more than 50% of cases, a user waits less than 60 seconds when the content is eventually delivered.

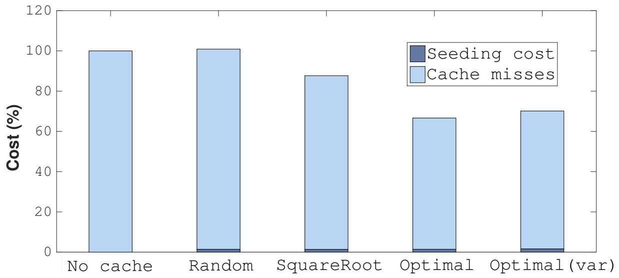


Fig. 9: Cost savings for different policies ($\Delta t = 23$ days).

VI. Related work

The high skewness in the content popularity and the rapidly increasing demand for mobile data has led to a number of recent papers related to mobile data offloading through opportunistic communications [9] [21] and distributed storage [14] [19]. For instance, in [9] the authors find a way to serve user requests from other mobile devices located geographically close to the user. They achieve this by clustering crowded places (*dataspots*), where a user can ask for contents directly to the close smartphones. [21] uses opportunistic local connections and remote connections through cellular networks. The solution proposed in [14] disseminates contents for Just-in-Time streaming (Netflix, Hulu or YouTube) in the access points distributed city-wide. [19] uses small cell access points (*helpers*) to cache contents. In their work, the authors allocate files in the helpers according to the network topology and the file popularity distribution. Files not available from helpers are transmitted by the cellular base station. In the context of vehicular cloud, some preliminary attempts to build an infrastructure similar to our proposal have been done by [37] and [38]. In these works, authors model an architecture to carry and forward contents making use of the predictable vehicle mobility [38] or propose a P2P content sharing scheme based on popularity [37], using a sub-optimal policy to store contents.

In this paper, we propose a new alternative caching infrastructure allowing to store popular contents in a vehicular cloud overlapping the existing cellular network; compared to other solutions, we limit resources constraints (e.g. caching in mobile devices) or we decrease installation and maintenance costs (e.g. caching in small cells). Moreover, in this work we use the high mobility of the vehicles in order to provide better performances and high availability of the popular contents. Finally, refreshing the caches allows to further exploit the variability of the content popularity to decrease the number of the accesses to the infrastructure.

VII. Summary and future work

Offloading contents is considered a solution to the rapid increase of mobile data demand and we firmly believe that vehicular networks will play an important role in this field. This is confirmed by the interest of research and companies in exploiting vehicles to carry contents or to give connectivity to users [2]. In this paper, we provide an analytical framework to allocate popular contents in a vehicular cloud. Based on our model, we suggest the periodicity of refreshing the caches in order to face the variability in the content popularity.

As a part of future work, we plan to extend our model including heterogeneity in content size, V2V communications and *location based* popularity (e.g. football videos are more likely to be downloaded near a stadium), using the recent studies on floating contents [35] [23]. We also plan to investigate the impact of different TTL per content to decrease the total load and/or to improve the end user Quality of Experience (e.g., analysing the slowdown metric).

References

[1] https://goo.gl/aMBZ7H.

[2] Veniam, Internet of Moving Things. https://veniam.com/.

[3] Your Car Is About To Get Smarter Than You. http://business.time.com/2014/01/07/your-car-is-about-to-get-smarter-than-you-are/.

[4] YouStatAnalyzer database. http://www.congas-project.eu/youstatanalyzer-database.

[5] H. Abrahamsson and M. Nordmark. Program popularity and viewer behaviour in a large tv-on-demand system. In *Proc. of ACM IMC*, pages 199–210, 2012.

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

[7] A. Antoniou and W.-S. Lu. *Practical optimization - algorithms and engineering applications.* Springer, 2007.

[8] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting mobile 3g using wifi. In *Proc. ACM*, MobiSys, pages 209–222, 2010.

[9] X. Bao, Y. Lin, U. Lee, I. Rimac, and R.R. Choudhury. Dataspotting: Exploiting naturally clustered mobile devices to offload cellular traffic. In *INFOCOM, Proc. IEEE*, pages 420–424, Apr 2013.

[10] Han Cai, I. Koprulu, and N.B. Shroff. Exploiting double opportunities for deadline based content propagation in wireless networks. In *INFOCOM, Proc. IEEE*, pages 764–772, April 2013.

[11] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *SIGCOMM, Proc. ACM*, pages 177–190, 2002.

[12] V. Conan, J. Leguay, and T. Friedman. Characterizing pairwise inter-contact patterns in delay tolerant networks. In *Autonomics, Proc.*, pages 19:1–19:9, 2007.

[13] R. Crane and D. Sornette. Robust dynamic classes revealed by measuring the response function of a social system. *Proc. of the National Academy of Science*, 105:15649–15653, October 2008.

[14] S. K. Dandapat, S. Pradhan, N. Ganguly, and R. R. Choudhury. Sprinkler: Distributed content storage for just-in-time streaming. In *CellNet, Proc. ACM*, pages 19–24, 2013.

[15] Z. Dezsö, E. Almaas, A. Lukács, B. Rácz, I. Szakadát, and A.-L. Barabási. Dynamics of information access on the web. *Phys. Rev. E*, Jun 2006.

[16] R. Durrett. *Probability: Theory and Examples*. Cambridge University Press, 2010.

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

[18] P. Gill, M. Arlitt, Z. Li, and A. Mahanti. Youtube traffic characterization: A view from the edge. In *SIGCOMM, Proc. ACM*, 2007.

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

[20] S. Ha, S. Sen, C. Joe-Wong, Y. Im, and M. Chiang. Tube: Time-dependent pricing for mobile data. In *SIGCOMM, Proc. ACM*, 2012.

[21] B. Han, P. Hui, V.S.A Kumar, M.V. Marathe, J. Shao, and A. Srinivasan. Mobile data offloading through opportunistic communications and social participation. *IEEE Trans. on Mobile Computing*, May 2012.

[22] Bo Han, Pan Hui, V.S.A. Kumar, M.V. Marathe, Jianhua Shao, and A. Srinivasan. Mobile data offloading through opportunistic communications and social participation. *Mobile Computing, IEEE Transactions on*, 11(5), May 2012.

[23] E. Hyytia, J. Virtamo, P. Lassila, J. Kangasharju, and J. Ott. When does content float? characterizing availability of anchored information in opportunistic content sharing. In *INFOCOM, Proc. IEEE*, pages 3137–3145, April 2011.

[24] T. Karagiannis, J.-Y. Le Boudec, and M. Vojnovic. Power law and exponential decay of intercontact times between mobile devices. *IEEE Trans. on Mobile Computing*, pages 1377–1390, Oct 2010.

[25] K. Lee, J. Lee, Y. Yi, I. Rhee, and S. Chong. Mobile data offloading: How much can wifi deliver? In *Proc. ACM*, Co-NEXT, 2010.

[26] R. Mahajan, J. Zahorjan, and B. Zill. Understanding wifi-based connectivity from moving vehicles. In *Proc. of ACM IMC*, pages 321–326, 2007.

[27] T. Mehmeti, F. Spyropoulos. Is it worth to be patient? Analysis and optimization of delayed mobile data offloading. In *INFOCOM, Proc. IEEE*, 04 2014.

[28] Z. H. Mir and F. Filali. LTE and IEEE 802.11p for vehicular networking: a performance evaluation. *EURASIP J. Wireless Comm. and Networking*, pages 89–89, 2014.

[29] A. Picu and T. Spyropoulos. DTN-meteo: Forecasting the performance of DTN protocols under heterogeneous mobility. *IEEE/ACM Trans. on Networking*, Feb 2014.

[30] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser. CRAWDAD data set epfl/mobility (v. 2009-02-24). Downloaded from http://crawdad.org/epfl/mobility/, Feb 2009.

[31] K. Poularakis, G. Iosifidis, A. Argyriou, and L. Tassiulas. Video delivery over heterogeneous cellular networks: Optimizing cost and performance. In *INFOCOM, Proc. IEEE*, pages 1078–1086, 2014.

[32] J. Robson. Small cell deployment strategies and best practice backhaul, August 2012.

[33] H. Schmidli. Lecture Notes on Risk Theory.

[34] P. Sermpezis and T. Spyropoulos. Not all content is created equal: Effect of popularity and availability for content-centric opportunistic networking. In *MobiHoc, ACM Proc.*, pages 103–112, 2014.

[35] N. Thompson, R. Crepaldi, and R. Kravets. Locus: A location-based data overlay for disruption-tolerant networks. In *Proc. of ACM CHANTS*, pages 47–54, 2010.

[36] M. Zeni, D. Miorandi, and F. De Pellegrini. Youstatanalyzer: A tool for analysing the dynamics of youtube content popularity. In *Proc. of the 7th International Conference on Performance Evaluation Methodologies and Tools*, pages 286–289, 2013.

[37] Y. Zhang, J. Zhao, and G. Cao. Roadcast: A popularity aware content sharing scheme in vanets. In *IEEE ICDCS*, pages 223–230, June 2009.

[38] J. Zhao and G. Cao. VADD: Vehicle-Assisted Data Delivery in Vehicular Ad Hoc Networks. In *IEEE INFOCOM*, pages 1–12, 2006.