Sprinkler: Distributed Content Storage for Just-in-Time Streaming

Sourav Kumar Dandapat, Swadhin Pradhan, Niloy Ganguly
Department of CSE, IIT Kharagpur, India
{sdandapat,swadhin.pradhan,niloy}@cse.iitkgp.ernet.in

Romit Roy Choudhury
Department of ECE and CS,
Duke University, USA
romit@ee.duke.edu

ABSTRACT

We envision cities where networking infrastructures, such as Wi-Fi access points (AP), will be equipped with storage capabilities. We propose to utilize the storage as a large distributed video cache. If successful, we envision that a child will be able to seamlessly watch a movie in a car, as her tablet downloads necessary parts of the movie over different Wi-Fi APs. The key challenge arises from the fact that the mobile tablet would not be able to download the entire movie from any single AP. Nonetheless, we show that the APs could be appropriately populated with video "chunks", such that the tablet can almost always get the needed chunk, *just-in-time* for video playback. Our system minimizes replication of video chunks, offering citizens with far greater number of videos to watch. We believe that such a video service could benefit cellular networks, by offloading their traffic to a sizable extent. This paper takes a first step into exploring such a city-wide content distribution service, and addresses one piece of the puzzle – efficient content storage.

Categories and Subject Descriptors
C.2.1[Network Architecture and Design]: Wireless communication
General Terms
Design
Keywords
VoD, Video Streaming, Network Cache Management

1. INTRODUCTION

Cellular data demands are escalating at a far greater pace than wireless network capacity. Given that wireless capacity is nearing Shannon's limit, researchers in academia and industry are looking for the next best solution. To this end, multiple studies are finding room for performance gains through modifications to the network architecture [7, 13, 19].

Of course, no single modification will cure the entire problem, rather, the evolving system will need to exploit every opportunity that comes along. These opportunities include, but are not limited to: (1) exploiting the proliferation of Wi-Fi access points through offloading; (2) leveraging advances in storage technology; (3) optimizing for video traffic; (4) improved data compression; etc. The design space is clearly vast and it is difficult for a single project to tap into all of them. This paper focuses on one "piece of the puzzle", wherein Wi-Fi APs, in a city, offload cell towers by posing as a distributed content storage system (utilizing attached storage with AP). The central goal is to allow mobile users download chunks of video from different APs, *just-in-time* for continuous playback. However, achieving the above goal is not trivial due to many reasons. For examples 1) association duration of a client device with an AP is not adequate to download a complete movie; so consecutive APs in path should be appropriately populated to facilitate downloading of the remaining part 2) different users may take different paths (source, destination, intermediate path); so continuous playback needs to be ensured independent of user's path.

One may argue that caching all the videos on all the APs will solve the above problem. Unfortunately, this limits the number of video choices under the "smooth streaming[1]" service to around $\frac{S}{V}$, where S denotes storage size at an AP, and V, the average size of a movie.

In addition, one may also think that a city where Wi-Fi AP density is adequately high does not need any such system – devices can connect to the remote server through the AP and download packets directly from them. In other words, the density needs to be such that the time taken to reach the next Wi-Fi AP must be less than the time over which the partially downloaded video can be played from the buffer. By bringing storage to Wi-Fi APs, and populating these storage appropriately, our system will allow much less AP density. Moreover, required download rate for continuous playback of high definition video may not be achieved when servers are far away.

Finally, this paper proposes a system called *Sprinkler*, in the spirit of how video content is carefully sprinkled across APs to achieve seamless streaming to mobile clients. Our key intuition is simple. Video chunks that need to be played earlier (e.g., initial scenes of a movie) are in greater need, while those which will be played later allow more time to download. This indicates that the availability of chunks with

CellNet'13, June 25, 2013, Taipei, Taiwan

---

[1] a list of videos being distributed through APs are available for "smooth streaming".

lower sequence number (initial chunks) should be higher. Seeded by this intuition, *Sprinkler* formulates an optimization problem that captures how different chunks have different deadlines, ultimately leading to an efficient scattering strategy. The formulation also aims to minimize the storage overhead, thereby allowing for more movie options to users. The main contributions may be summarized as follows:

- **Viewing multiple Wi-Fi APs as a "scattered cache" wherein each AP stores parts of the same content.** While distributed caches have been extensively studied, to the best of our knowledge, the notion of scattering the content in space considering user's mobility is relatively unexplored.
- **A scalable and flexible framework which is optimized to offer more number of videos.** The combined solution is simple and possesses desirable properties for scaling to larger networks, different mobility patterns, and various network densities.
- **A promising offloading gain when tested (through simulations) on real city maps, with realistic traffic patterns, wide range of speed and with various AP densities.** Under average conditions, *Sprinkler* facilitated over 85% (up to 65% in high traffic situation) offload from cellular network. Moreover, the performance stayed stable over a wide range of speed. Interestingly, its performance actually improved when in high traffic situation the mobile clients had to stop at various crossings and it also showed resilliance, in term of performance, in sparse AP distribution.

The rest of this paper expands on these contributions, beginning with some motivation and measurements, and followed by design and evaluation.

2. SYSTEM SETTINGS

This section describes the motivation, scope, and settings against which this work is positioned. We present measurement results to justify certain assumptions.

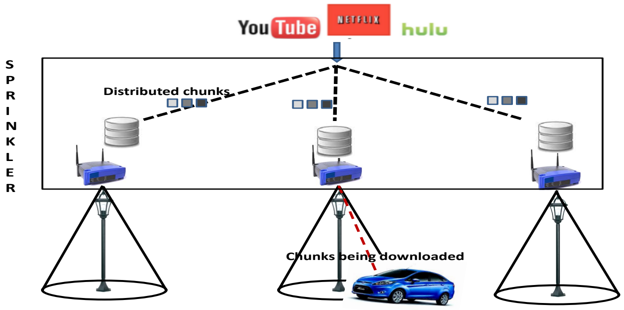

Figure 1: Video content is periodically pushed to Wi-Fi. Vehicles passing an AP downloads content from it thus offloading cellular traffic.

Fig. 1 shows the system setting of *Sprinkler*. *Sprinkler* has two functionalities - 1) pulling videos periodically from online services such as Netflix, Hulu etc. based on popularity, content category etc. and scattering different "chunks" of the videos to different Wi-Fi APs, 2) interacting and serving video "chunks" to mobile devices. When a device is in contact with a Wi-Fi AP, it attempts to download chunks that it will need in the future. These chunks need not be in order, i.e., if the device already has, say, chunks 1 to 5, and the Wi-Fi AP advertises chunks 1, 3, 7 and 12, the device attempts to download both 7 and 12. Of course, it is possible that the client disconnects from the AP before chunk 7 is downloaded in its entirety. In that case, the client caches the packets of chunk 7, but scans for another AP that has chunk 7 – on encountering one, it completes downloading the remaining packets. On the other hand, if downloading of chunk 7 is complete, the client proceeds to download the next available chunk (12 in this example). Clearly, if a client is waiting on a traffic light, the connection opportunity with the AP is fully utilized. In this manner, *Sprinkler* hopes that the device will download chunk i before it needs it for playback. If, however, the needed chunk is not available, it switches to cellular network (It is assumed that mobile devices in vehicles have access to both Wi-Fi and 3G/LTE, and can choose to multiplex between the two.) and continues streaming the video.

A few questions are natural to ask against the above setting:

(1) Why is it necessary to locally store content on the APs? Why not "pull" content directly from remote servers when a mobile client requests it? We observe that if a static client requests a video, it makes complete sense to download the content from the server and stream it over Wi-Fi. However, if the client is mobile (i.e., likely to stay connected to the Wi-Fi AP for only a few seconds), then time is of the essence. Storing the video on the AP enables much faster download, compared to the far-away server.

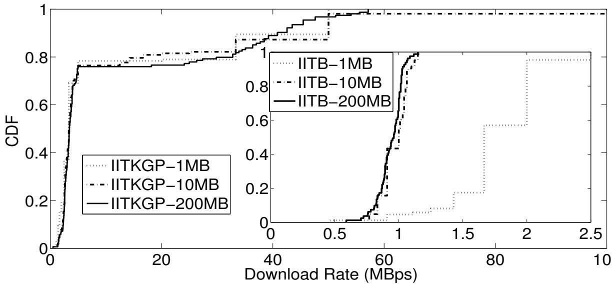

Figure 2: Change of download rate with file size when file is hosted in local server (IIT Kharagpur, India) and inset shows the rate when file is hosted in IIT Bombay, India.

To quantify this, we measured file download rate through mobile Wi-Fi client from the different servers located at different distances (local: IIT Kharagpur, India, and in a different city (2000km away): IIT Bombay, India). Fig. 2 shows that the download rate is about 8 times higher when the file is downloaded from local server. Moreover, download rate deteriorates with increase in file size if the file is hosted far away, while the rate of change is relatively less when file is hosted locally. As duration of download increases with file size, the uncertainties of the network lead to the deterioration of download performance from remote server.

(2) Instead of storing video at every AP if we make local hubs (may be APs in junctions) to store video then can we get equivalent performance as Sprinkler? To serve video to

clients, non-hub APs need to fetch the required video from its nearby hub. Though there is wired connection between a non-hub AP and a hub AP, it takes different amount of time to fetch the video from hub depending on the traffic in wired network.

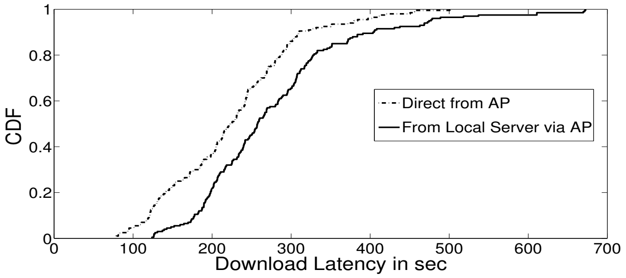

Figure 3: Two scenarios of downloading a file of size 100MB : 1) file is hosted in AP itself 2) contacted AP fetches file from another AP.

To quantify this measure we have done an experiment where two scenarios are considered, in one scenario, client requests to a hub-AP, i.e., download videos directly from an AP which has storage and in another scenario, client requests to a non-hub-AP, i.e., indirect download through requested AP. Fig. 3 shows the performance achieved in the two scenarios in terms of download latency during downloading a file of size 100MB. There is consistently a performance gap of more than 15% in terms of download latency which motivates us to store videos in every AP instead of APs in junctions.

The next section presents the design of *Sprinkler*, followed by its evaluation in Section 4.

3. SPRINKLER: SYSTEM DESIGN

This section begins with a high level overview of *Sprinkler*, followed by the specific problem formulation and design.

3.1 Overview

We envision *Sprinkler* APs to be atop city lamp-posts, rooftops, or public bus stops. Each AP maintains an index of the list of movies and their respective chunk IDs cached in its storage. When a mobile client (watching / planning to watch a movie) associates to an AP, it checks whether the AP has any of the chunks that it does not have and would need in the future. If so, the client requests downloading the chunk with the least sequence number in the AP's cache, and not present with client.

3.2 Chunk Distribution Intuition

An efficient strategy, for chunk distribution, can be developed from the following intuition. Availability of initial chunks should be higher [14]. A user starting to play a movie will need the initial portion of the movie – the first chunk – right away. Since the user can start watching at any location, the first chunk needs to be available at every AP. However, the 2^{nd} chunk needs to be downloaded by the time the 1^{st} chunk has played out. More generally, let us assume that a user moves past k APs during the viewing time of a single chunk. So, user will start viewing chunk X after she crosses $(X-1)\times k$ APs, then we have to ensure the availability of chunk X within $(X-1)\times k$ consecutive APs. Extending this logic, at least one copy of chunk $X+1$ should be available within $X\times k$ consecutive APs from the starting location. Thus, higher numbered chunks can be made available at proportionally less frequency – the ratio of availability for chunks 1:2:3, can be modeled as $1{:}\frac{1}{k}{:}\frac{1}{2k}$. Of course, this model suits a simple situation where APs are equally spaced out and organized on a straight line. However, *Sprinkler* handles arbitrary road networks and AP positions, and the system gets more involved.

3.3 Formulating as a Linear Program

Building on the above intuition, we formulate the general problem as a linear program. We require as input a topology of APs, and a city specific *ideal speed* at which on an average car moves in that city. We assume that users are moving on the shortest path between their source and destination (i.e., no loops exist in their motion path). The details follow.
Let us assume X_1, X_2,...,X_p represent p consecutive APs in a shortest path of a city; and n, the number of APs in the network and k is the number of APs, a client can cross within viewing time of a chunk. Let an indicator variable z_i^j be associated to AP X_i, where $z_i^j=0$ signifies the absence of the j^{th} chunk at AP X_i and $z_i^j=1$ signifies its presence. Availability of first chunk at every AP can be trivially expressed as below:

$$z_i^1 = 1 \tag{1}$$

To ensure that the j^{th} chunk is present at least once in a path (X_i, ..., $X_{i+(j-1)k}$) of length $(j-1)\times k$, the following condition must be satisfied:

$$z_i^j + z_{i+1}^j + ... + z_{i+(j-1)\times k}^j \geq 1 \tag{2}$$

where i varies from 1 to $p-k\times(j-1)$ and j varies from 2 to m (number of total chunks in a movie file). We will get a set of constraint equations considering all shortest paths and all chunks. For ease of storage management, every AP should have a maximum (β) as well as a minimum limit (θ) on the number of chunks that can be kept in one AP. Mathematically, these constraints can be expressed as follows.

$$\theta \leq \sum_{j=1}^{m}(z_i^j) \leq \beta, \forall i = 1,\dots,n \tag{3}$$

Our objective function is to minimize total storage (f_{obj}). f_{obj} is formally expressed as below:

$$f_{obj} = \sum_{i=1}^{n}\sum_{j=1}^{m}(z_i^j) \tag{4}$$

This optimization problem can be solved using Integer Linear Programming (ILP), where the number of variables equal $n\times m$. We use LPSOLVE [1] for solving the ILP – the package outputs the assignments of chunks for each AP.

4. EVALUATION

In this section we present a detail evaluation of the performance of Sprinkler. We provide a detail description about the experimental set-up and parameters. We also describe metric of our interest for evaluation. We also compare its performance with baseline schemes which are described next.

4.1 Experimental Setup and Simulation

Network Simulator and Parameter: We perform simulation based experiments using the NS3 simulator. We use constant data rate (9Mbps - 24Mbps) model for our experiments. We have used Jake's propagation loss [15] model to realize an urban environment.

Road network and AP Placement: We use the road network of a part of Mysore (Fig. 4), a city 130km away from Bengaluru, as a case study. Incidentally Mysore is also the first Wi-Fi enabled city of India as well as the second Wi-Fi enabled city of the world [5]. In the simulation, we digitize this road network and place Wi-Fi APs – we first place APs at all traffic intersections, and then place additional ones at every z meters (value of z is taken as 100 / 150 / 200 / 250 / 300 for different simulation scenarios).

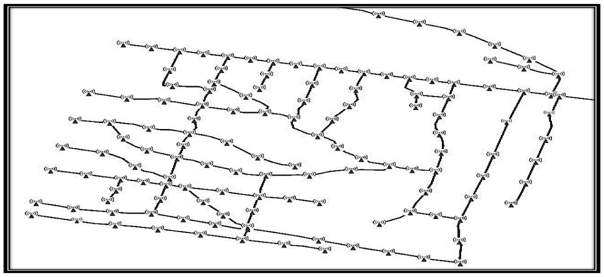

Figure 4: Road Map of Mysore with AP placement (first Wi-Fi enabled city of India).

Client's Mobility: The clients move between randomly chosen source and destination locations on the map, and follow the shortest route between them – their speeds vary, and they pause at different traffic intersections for a random duration. Clients either move in low traffic scenario, where they rarely come together or they move in high traffic scenario.

Choice of chunk size: Videos and movies are broken into chunks, such that at least one chunk may be downloaded while a mobile user passes through a roadside Wi-Fi AP with *ideal speed*. We assume there is a city specific *ideal speed* on which maximum vehicles move and also assume that the length of the road on which Wi-Fi is available is twice the transmission radius of Wi-Fi (assumed 100m). Finally, assuming that the minimum data rate can be 9 Mbps base rate, we can compute the size of each chunk (3MB in this paper). In reality, if the car is moving slower, or if the data rate is higher when the car is just below the AP, more than one chunk can be downloaded.

Chunk Distribution: For our experiments, chunks are distributed considering that a car moving at 40km/ hour (as *ideal speed*) gets its desired chunk within time (see section 3.2). Throughout the experiments, chunk size is assumed to be 3MB. A video is assumed to be comprised of 30 chunks and every AP can store at most 10 chunks of each file. The APs are populated as prescribed by the ILP.

Video Playback Rate: Uninterrupted viewing depends on download rate experienced by client as well as consumption rate of video player. For our experiments, we assume playback rate for the video is 700MB/hour.

***Far-Sprinkler*:** It is a system where the APs don't locally host video, but pull them from distant servers. This is used as a scheme for comparison. Download rate that a client experiences in this environment depends on the relative locations and distribution of servers from client. So, this download rate is likely to vary across countries even across different cities of a country. *Far-Sprinkler(x,y)* will denote x% of servers are nearby and y% of servers are far away. We have considered three scenarios for different (x, y) value sets - (80, 20), (70, 30) and (60, 40). However, actual download rate experienced by a *Far-Sprinkler* client might be significantly less. For example in the case of India, with respect to Tweet traffic, 36% of traffic is from within the country (similar as local server) and 64% of traffic is from outside (similar to far away server)[18].

4.2 Metrics of Interest

Our goal is to provide an uninterrupted video playback service to mobile clients with least possible switching to other networks at different AP densities. In view of this, we will evaluate *Sprinkler* across 2 main metrics:

1. **Fraction of data offload (FDO)** measuring the percentage of the video packets during the vehicle's journey downloaded over Wi-Fi. We will particularly look into the difference between *Sprinkler's* FDO and FDO from the Basic scheme (where packets are directly downloaded from the server).

2. **Cost efficiency** will capture FDO gained per AP. For example, with X number of APs, if the FDO of a scheme is Y then cost efficiency of that scheme is Y/X.

4.3 Simulation Results

In this subsection, we present all our simulation results and detailed analysis of those results. Specifically we present results of the effect of 1) AP density (varies with inter-AP distance) 2) speed of the vehicle 3) traffic on the performance of *Sprinkler* system.

Effect of AP Density: In this experiment, for different inter-AP distances (100m/150m/200m/250m/300m), we have measured the *Sprinkler's* performance in terms of Fraction of Data Offload (FDO) and cost efficiency. It is assumed that cars are moving in a speed range of 30km/hour - 50km/hour. Base data rate is assumed as 9Mbps.

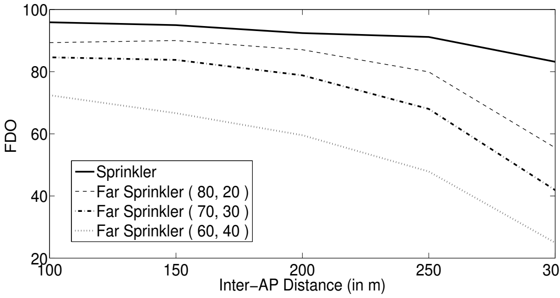

Figure 5: Fraction of data offload with different inter-AP distance. The data rate is assumed as 9Mbps and chunk size is assumed to be 3MB.

Fig. 5 compares the FDO of *Sprinkler* with different *Far-Sprinkler* schemes with respect to different inter-AP distances. *Sprinkler's* performance in terms of FDO is over 90% even when the inter-AP distance is 250m. For *Sprinkler* the performance degrades slowly and gracefully while the degradation is quite sharp for *Far-Sprinkler* schemes. Even with 300m inter-AP distance, *Sprinkler* offloads more than 85% of data while *Far-Sprinkler(*80,20*)* offloads around 60%, *Far-Sprinkler(*70,30*)* offloads around 45% and *Far-Sprinkler(*60,40*)* offloads just 25% of data. It implies that with much lesser number of AP, *Sprinkler* can offload a significant portion of traffic from 3G or cellular network.

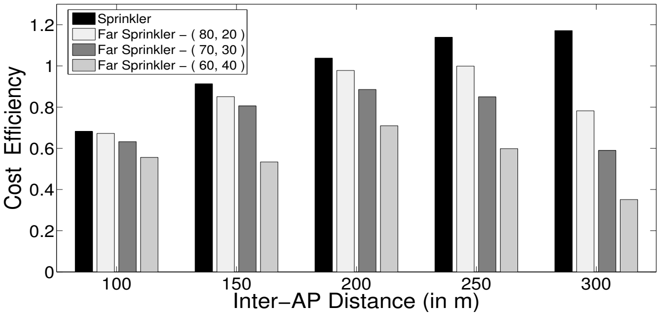

Figure 6: Cost efficiency with different inter-AP distance.

Fig. 6 compares cost efficiency of *Sprinkler* with different schemes of *Far-Sprinkler.* At lower inter-AP distance (where there is virtually blanket coverage), all schemes are almost equally cost-efficient. As inter-AP distance increases the difference in cost efficiency among *Sprinkler* and different schemes of *Far-Sprinkler* increases.

Effect of Vehicle's Speed: We measure the percentage of average data offload from 3G using *Sprinkler*, while the vehicle moves at different *average speeds.* In this experiment, a set of clients move with speed randomly chosen from a range of $(x - 10)$ to $(x + 10)$ km/hour such that the average speed of those clients become x. We compare *Sprinkler* with *Far-Sprinkler(*80,20*)* and *Far-Sprinkler(*70,30*)*. It is assumed that APs are placed at every 100m.

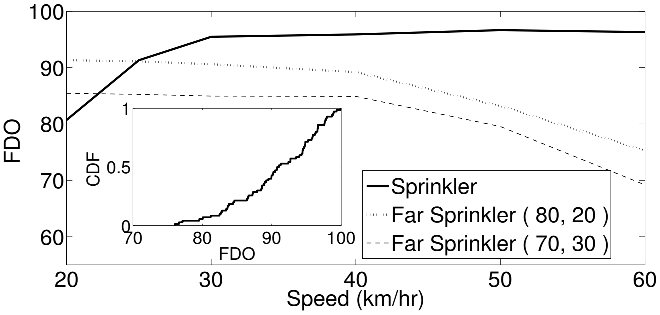

Figure 7: FDO against speed while cars move at different speeds and APs are placed at every 100m. Inset shows CDF plot of FDO while cars move at random speed in between 30Km/hour - 50Km/hour.

From Fig. 7 it is evident that *Sprinkler's* data offload performance is over (90%) across a wide range of speed. When a client moves at lower than ideal speed, it remains associated with an AP for longer duration (> required time to download a chunk). Longer association duration becomes effective for client which can then download multiple consecutive chunks required for near future viewing. But a client with low speed takes longer duration to move to the next AP. In such a scenario, the client, may have to switch to other network, if it fails to download multiple consecutive chunks from AP. However, presence of consecutive chunks is not guaranted by chunk distribution strategy. So, at lower than ideal speed, fraction of data offload becomes comparatively low. On the contrary, at lower speed, *Far-Sprinkler* client performs better as it remains associated with an AP for longer duration and there is guarantee of receiving chunks in sequence. FDO reaches its maximum value ≈ 96% when we consider *ideal speed.*

To verify the robustness of *Sprinkler*, in terms of FDO, across a wide range of speed we have done an experiment where car moves at a random speed chosen in the range of 30 km/hour - 50 km/hour. Inset of Fig. 7 shows the CDF plot of FDO of the above mentioned experiment. It shows that over 60% of times client's FDO is more than 90% and FDO varies between 76 to 100.

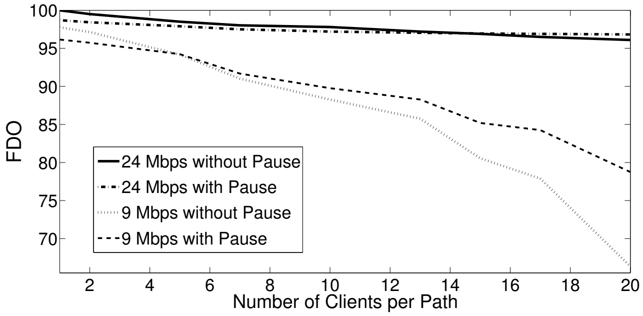

Figure 8: Data offload performance of Sprinkler with presence and absence of pausing time against different traffic load with various AP data rate.

Effect of Traffic: In this experiment we have considered different traffic loads. We have assumed that initially, each single user starts travelling and after some steps, X (= 1, 5, 10, 15, 20) number of users come together and travel through a designated path. We have considered two situations, one where the vehicles don't have to stop at crossings and second where they have to wait at 20% of crossings for a random time; the pause time instances are drawn from a distribution with a mean of 3 seconds. All the users compete with each other to get their slice of bandwidth from the APs along the path. We have done this experiment assuming different download rates of AP (9Mbps, 24Mbps). Fig. 8 shows that for the Sprinkler system performance degrades gracefully under traffic. However, more interestingly, the performance of *Sprinkler* is much better (e.g. ≈ 80% for 9 Mbps traffic load = 20) when the cars have to stop at the crossing. This happens because, the clients while waiting in the crossing get enough time to download all the chunks from the AP they are connected to. The more the clients, more is the opportunity to download the contents of the AP they are connected to. Hence, the percentage improvement is even better as the traffic becomes heavier and the data rate is low. In a realistic setting, the amount/number of times a vehicle has to wait in the crossing will be directly related to the density of traffic present in the area. Thus, this result establishes the potential of *Sprinkler* in realistic situation.

5. RELATED WORK

The Internet has witnessed the sharp increase of video traffic in the recent years with all kinds of Internet streaming systems, such as VoD and Internet P2P-based streaming systems [9] etc. And with the huge surge in smartphone, more and more accesses from mobile devices are directed to all kinds of Internet streaming services. Almost all popular streaming service providers, including Youtube [4], Netflix [2] and Hulu [3], also provide streaming services to subscribed mobile users via APPs built in various mobile operating systems. This is virtually pushing the capacity of 3G to a limit. An important and not-so-new proposal to handle the situation is to devise techniques to push 3G traffic to Wi-Fi network [7]. The users also can derive several benefits from it, for example, typically a smartphone accessing content through Wi-Fi would have its energy drained at a much lower rate than it would be through 3G [8], the pricing for data download which is still being widely discussed and debated [16] would certainly go down. However, there are flip sides in trying to offload streaming services to Wi-Fi. Wi-Fi although when connected, indeed delivers high throughput even in a mobile scenario, but has frequent disconnections even in a commercially operated, metro-scale deployment [10]. Hence, the nature of offload has primarily being directed towards offloading delay tolerant services [11].

Side by side there have been several efforts to (a). make the video streaming lightweight/efficient to combat the problems arising out of Wi-Fi channel uncertainties. [20], (b). improve the Wi-Fi protocol to ensure seamlessness. Due to mobility, typically duration of association with an Wi-Fi AP is very short, and hence, protocols to ensure fast association need to be established. An important attempt in this direction is the proposal of Cabernet Transport Protocol (CTP) [12] which incorporates fully automatic scanning, AP selection, association, DHCP negotiation, address resolution, and verification of end-to-end connectivity, as well as detection of the loss of connectivity. Authors of [17] propose a smart, energy-efficient and fast way to detect and connect Wi-Fi. [10] investigate a transport layer protocol design that integrates 3G and Wi-Fi networks, specifically targeting vehicular mobility. The goal is to move load from the expensive 3G network to the less expensive Wi-Fi network without hurting the user experience.
There has been also important works [7] to ensure fast switching to 3G in order to overcome the poor availability and performance glitches of Wi-Fi. All these efforts are improving the reliability of Wi-Fi and making it suitable for streaming applications. For example, new standards 821.11e and 802.11p, 802.11-2012 [6] has been proposed which ensures QoS and mobility respectively. In light of such developments, we believe our proposal of carefully placing content directly into APs is timely and would enhance its usability in offloading streaming content.

6. CONCLUSION

We envision future smart cities will provide a video streaming service to mobile users – "a movie theater in my car". With cellular network capacity drying up, such a service may be difficult to support over 3G/LTE connections. On the other hand, the proliferation of cheap storage technology can enable new architectures where network infrastructure (such as Wi-Fi APs) is equipped with content storage. We design *Sprinkler* to take advantage of such nearby storage and scatter content in them such that *just-in-time* streaming can be supported. This paper is an early step in this direction with much more research remaining to make it an end to end reality.

7. REFERENCES

[1] http://lpsolve.sourceforge.net/5.5/.
[2] https://www2.netflix.com/mobile/.
[3] http://www.hulu.com/plus.
[4] http://www.youtube.com/mobile/.
[5] Mysore, first wi-fi enabled city in India, http://en.wikipedia.org/wiki/wi-fi.
[6] New wi-fi standard means better vehicle support, improved cell handoff, http://gigaom.com/mobile/new-wi-fi-standard-means-better-vehicle-support-improved-cell-handoff/.
[7] A. Balasubramanian, R. Mahajan, and A. Venkataramani. Augmenting Mobile 3G Using WiFi. In *MobiSys*, pages 209–222, San Francisco, California, USA, June 2010. ACM.
[8] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy Consumption in Mobile Phones: A Measurement Study and Implications for Network Applications. In *IMC*, pages 280–293, Chicago, Illinois, USA, November 2009. ACM.
[9] H. Chi, Q. Zhang, J. Jia, and X. (sherman) Shen. Efficient search and scheduling in P2P-based media-on-demand streaming service. *JSAC*, 25:119–130, 2007.
[10] P. Deshpande, X. Hou, and S. R. Das. Performance Comparison of 3G and metro-scale WiFi for vehicular network access. In *IMC*, pages 301–307, Melbourne, Australia, November 2010. USENIX.
[11] S. Dimatteo, P. Hui, B. Han, and V. O. Li. Cellular Traffic Offloading through WiFi Networks. In *MASS*, pages 192–201, Valencia, Spain, October 2011. IEEE.
[12] J. Eriksson, H. Balakrishnan, and S. Madden. Cabernet:Vehicular Content Delivery Using WiFi. In *MobiCom*, pages 199–210, SanFrancisco, California, USA, September 2008. ACM.
[13] B. Han, P. Hui, V. A. Kumar, M. V. Marathe, G. Pei, and A. Srinivasan. Cellular Traffic Offloading Through Opportunistic Communications: A Case Study. In *CHANTS*, pages 31–38, Chicago, Illinois, USA, September 2010. ACM.
[14] K. A. Hua and S. Sheu. Skyscraper broadcasting: a new broadcasting scheme for metropolitan video-on-demand systems. In *SIGCOMM*, pages 89–100, Cannes, France, October 1997. ACM.
[15] W. C. Jakes and D. C. Cox, editors. *Microwave Mobile Communications*. Wiley-IEEE Press, 1994.
[16] C. Joe-Wong, S. Ha, and M. Chiang. Time-Dependent Broadband Pricing: Feasibility and Benefits. In *ICDCS*, pages 288–298, Minneapolis, Minnesota, USA, June 2011. IEEE Computer Society.
[17] K.-H. Kim, A. W. Min, D. Gupta, P. Mohapatra, and J. P. Singh. Improving Energy Efficiency of Wi-Fi Sensing on Smartphones. In *Infocom*, pages 2930–2938, Sanghai, China, April 2011. IEEE.
[18] J. Kulshrestha, F. Kooti, A. Nikravesh, and K. P. Gummadi. Geographic Dissection of the Twitter Network. In *ICWSM*, Dublin, Ireland, June 2012. AAAI.
[19] Y. Li, G. Su, P. Hui, D. Jin, L. Su, and L. Zeng. Multiple Mobile Data Offloading Through Delay Tolerant Networks. In *CHANTS*, pages 43–48, Las Vegas, Nevada, USA, September 2011. ACM.
[20] Y. Liu, F. Li, L. Guo, B. Shen, and S. Chen. A Server's Perspective of Internet Streaming Delivery to Mobile Devices. In *Infocom*, pages 1332–1340, Orlando, Florida, USA, March 2012.