Managing Massive Firmware-Over-The-Air Updates for Connected Cars in Cellular Networks

Carlos E. Andrade, Simon D. Byers, Vijay Gopalakrishnan, Emir Halepovic,
Milap Majmundar, David J. Poole, Lien K. Tran, Christopher T. Volinsky
AT&T Labs
Bedminster, NJ, United States

ABSTRACT

We consider the problem of managing Firmware Over-The-Air (FOTA) updates for cars at a massive scale over a cellular network. This problem is constrained by factors like large number of cars, a need to perform updates quickly and securely, ability to do the update anywhere and any time, and delivering the often large update without harming the network. We present a scheduling approach for managing large FOTA downloads in a cellular network that combines historical network load with car usage and location analytics. The proposed approach uses a combination of heuristics and an optimized scheduler and applies them to schedule the admission of cars to download FOTA. Using real network data from a large cellular provider that includes a million cars and nearly a billion radio-level network connections, we show that our scheduling approach is feasible and practical. We also show that it is possible to manage FOTA even when device and network models use high levels of aggregation over time. Simulation results show that our method improves over random uncontrolled approaches by (i) reducing median download startup delay by 48%, (ii) reducing the number of cars that don't complete the update by 10% and most importantly, (iii) reducing the load in busy cells by up to 37%.

1 INTRODUCTION

With an estimated 100 million lines of software code residing in various systems throughout a vehicle [32], it is no exaggeration that cars today are increasingly reliant on and controlled by software. Software powers not only the in-car infotainment, or the critical safety sensors such as airbag sensors, speed, electronic stability, and collision prevention, but also the behavior of core components like brakes, steering and suspension. The use of cellular connectivity to support emergency response, in-car communications and WiFi hotspot, and entertainment makes connected cars a fast growing segment of Internet of Things (IoT); forecasts predict that 90% of cars will be connected by the year 2020 [28].

The software in cars, like any software, requires updates to address bugs, security vulnerabilities or enhance the cars' capabilities. For example, software-only recalls affected several million vehicles in the U.S. in 2015 [22]. With an estimated per-service cost of roughly $100 [24], the traditional way of applying software updates at a dealership is both time-consuming for manufacturers, as well as expensive. It is also a major inconvenience for the car owners.

Naturally, given the critical role of software and the enhanced levels of regulation on safety recalls for cars to other consumer devices, manufacturers (and owners) are looking for ways for timely, reliable, and effective software updates. A leading contender is the ability to updated firmware over-the-air (FOTA). While these are sometimes referred to as Software OTA (SOTA) for infotainment and other non-critical functions and Firmware OTA (FOTA) for mission-critical systems of the car, we refer to them jointly as FOTA in this paper for brevity. With FOTA, a device downloads the update via its network interface (cellular [5] or WiFi [6]) and updates itself. FOTA offers a safe, convenient, and cost effective method to deliver software updates to vehicles because it can scale and does not require physical access to the car. Studies suggest that automakers will save $35 billion by 2022 [19] by adopting FOTA.

While FOTA offers benefits to automakers and owners, it presents a challenging problem to the cellular network operator. Depending on the type of update, these updates can range from tens of Megabytes to even Gigabytes [8]. These large FOTA updates come in addition to already exploding multimedia and other types of throughput intensive traffic. Worse yet, as we show in Sec. 2.2 using detailed radio-level simulations, these updates have the potential to significantly interfere with experience of regular users. Hence the cellular network operator needs to identify mechanisms that allow automakers to conduct successful FOTA campaigns, while at the same time ensure the quality of service to all other network users.

In theory, there are several ways one could address this challenge. Some manufacturer allow owners to use a USB drive to perform updates. This, however, only makes sense for a tech-savvy owner. Another approach is to use WiFi for FOTA and indeed some manufacturers (plan to) use WiFi for their updates [6]. However, we envision that cellular-based OTA updates will dominate OTA updates because of its coverage and simplicity. Other alternatives are to time-shift the FOTA download to off-peak times, or pre-fetching data opportunistically [11, 15, 17, 27]. These approaches, however, are not always applicable to FOTA due to the current constraints of the deployed network technologies and radio device implementations in cars. For example, most cars can connect to the network *only when their engines are running*. Finally, many of the today's cars have limited capabilities in terms of interactions with the network and servers; hence approaches that rely on intelligent coordination between the device and the network are not feasible, such as availability signaling or pausing under high load.

In this paper, we propose to use a combination of analytics and intelligent schedule optimization to manage FOTA over cellular

CarSys'17, October 20, 2017, Snowbird, UT, USA.
© 2017 Association for Computing Machinery.
ACM ISBN ISBN 978-1-4503-5146-1/17/10…$15.00
https://doi.org/10.1145/3131944.3131953

networks. Specifically we study car location, usage patterns, as well as network load to come up with a schedule for cars to download their updates. Our approach, described in Sec. 3, uses radio-level network data to derive car mobility patterns across time and space. We use this information to compute the admission schedule for FOTA. Our schedule optimizer (based on Biased Random-Key Genetic Algorithm, or BRKGA) tries to balance successful completion of updates in an allotted time, with the increased network load due to FOTA. The problem of computing a schedule for millions of cars is not only challenging due to its scale, but also because of real-world challenges (e.g., new cars, infrequently used cars, random patterns). To address this, we use problem reduction and parallelization. Specifically, we reduce the problem space by first segmenting cars into groups and using heuristics to schedule some groups. We then compute the schedule for the remaining groups using BRKGA framework by dividing cars based on geographic regions and computing regional schedules in parallel.

Using simulations based on network data from a large cellular provider we compare our approach with random, uncontrolled approach in Sec. 4 to show the comparative reduction in network impact due to scheduling. Specifically, we show that our approach reduces median download startup delay by 48% compared to random, reduces the number of cars that don't complete the update by 10% and most importantly reduces the load in busy cells by up to 37%. These results demonstrate that our approach is not only successful at reducing the impact of FOTA on the network, but is also scalable and practical.

2 MOTIVATION

While OTA updates of cars appears to be the future, the networking technology to deliver the update is less clear. In this section, we first motivate why we believe that cellular is likely to become the dominant technology for FOTA delivery. We then articulate the projected impact of FOTA on cellular networks and discuss why many of the obvious techniques are insufficient.

2.1 Why cellular networks for FOTA?

While there are many network technologies — WiFi, Cellular, Bluetooth, Satellite — that can be used for FOTA, there are both technical and non-technical reasons that favor cellular networks as the preferred "medium" for FOTA delivery and as a technology that is actually driving the OTA update trend [18].

Car manufacturers are going to consider multiple objectives when selecting the right networking technology for FOTA updates. In particular, they will want to minimize costs or changes (avoid additional hardware or intelligence), require minimal effort from owners, exert control over the roll-out of updates (FOTA campaign), and most importantly require roll-out (and accounting) of updates to all affected cars in a timely manner.

The only medium that addresses all these criteria today is cellular. Manufacturers are already adding cellular connectivity to cars [29] to offer a WiFi hotspot, support infotainment, emergency response and enable telematics. As a technology that is *already in place*, it does not require additional hardware or a change to the production process. The reach and widespread availability of cellular networks allows for *any time, anywhere* updates; this is going to grow in importance as cars become more software-driven and autonomous. Just-in-time updates are already seen as a key feature according to market research [18]. With cellular, manufacturers can deliver updates to cars anywhere; in cities or suburbs, highways, in driveways, open parking lots or even parking garages. They can have full *control* path using cellular network. In fact, manufacturers and third party vendors are announcing or already delivering FOTA and FOTA management solutions that use cellular networks [6, 18].

One of the key concerns today with cellular networks, however, is the cost of data. As a result some manufacturers [6] are looking to WiFi for FOTA. WiFi, however, requires that (a) the car have a WiFi module that can communicate with the network — which most cars do not have, and (b) the car owner set up and maintain WiFi connectivity to the car. While that may be possible for a subset of cars (e.g., those parked at home), there is no guarantee that all cars will ever get in the range of a stable and convenient WiFi network (e.g., cars owned by downtown high-rise residents that are typically parked in underground garages). Satellite networks offer high-bandwidth and wide coverage, but again require additional hardware and don't often offer adequate control. Techniques like downloading to a smartphone via a specialized app and then updating via Bluetooth [19], using USB sticks, are all viable alternatives, but require human intervention to ensure the update is completed. This makes it cumbersome and time consuming for owners, and prevents the manufacturers from ensuring that all affected cars are updated. For all these reasons, we believe that the alternatives will become either niche or special cases [19].

2.2 Impact of FOTA on cellular networks

Having motivated that cellular is the primary candidate to deliver FOTA, we now examine the impact of FOTA on cellular networks. The impact of FOTA traffic can be considered across multiple dimensions. One dimension is the immediate or short-term impact on the total load inside the cell and surrounding cells. This impact is reflected in how FOTA affects performance of other users and their own performance, as well as overall radio resource utilization and efficiency, both in the cell that delivers FOTA and surrounding cells. Another dimension is the long-term impact, reflected mainly in capacity planning due to this type of traffic, as significant increase in busy hour traffic may trigger capacity upgrade. Both dimensions apply to all segments of the network, including radio access or last mile, backbone and core capacity, including links and network elements like routers and gateways. We focus on short-term impact of FOTA, since mitigating problems in this dimension will soften long-term issues.

2.2.1 Impact on PRB Utilization. In cellular LTE networks, a unit of radio resource is a Physical Resource Block (PRB). PRBs have both a time and frequency domain, i.e. a duration of 1 ms and a bandwidth of 180 kHz. There is a fixed number of available PRBs depending on the total carrier bandwidth. PRB utilization level, U_{PRB}, is typically expressed as a percentage and is used to represent the load level of the cell. Studies suggest that it is desirable to keep U_{PRB} under 80% to maintain good performance for all users; otherwise users' wait time for content increases sharply [7].

Even a single user can momentarily saturate the uplink or downlink capacity of a cell with a data burst that takes up all PRBs.

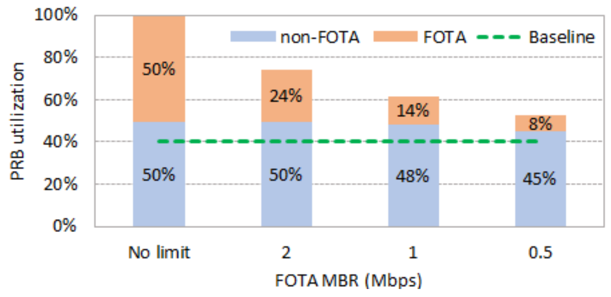


Figure 1: Impact of FOTA at different MBR in the same cells.

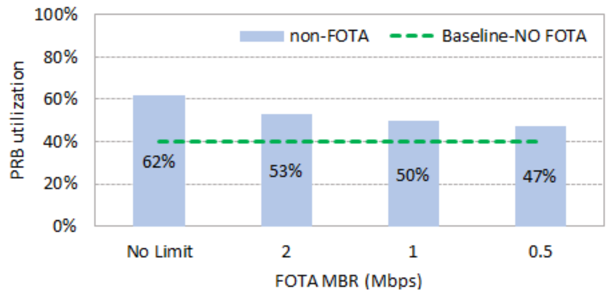


Figure 2: Impact of FOTA at different MBR in other cells.

While this may not always cause disruption to user experience, large transfers (100s of MB) sustained for long time periods can result in degraded performance for other users.

Using a custom, highly detailed, LTE system-level simulator, we studied the impact of FOTA on PRB utilization. We simulated ten seconds of traffic in 21 cells, 7 of which had one or two simultaneous FOTA downloads. The non-FOTA traffic consisted of 150 KB downloads with exponential arrivals. The baseline non-FOTA load was set to 40% utilization level. FOTA traffic was delivered as persistent bulk download at four separately simulated rate limits or Maximum Bit Rate (MBR) regimes: unlimited, 2, 1, and 0.5 Mbps. We use a channel bandwidth of 10 MHz, carrier frequency of 740 MHz, inter-site distance of 500 meters, HATA Urban propagation model and log-normal shadow fading in our simulations.

The impact on U_{PRB} in cells with FOTA is shown in Figure 1. As expected, FOTA downloads saturate capacity when not rate-limited, and add varying amounts of load depending on the rate-limit. FOTA impact extends to adjacent non-FOTA cells also due to increased interference. Figure 2 shows that the impact on adjacent cell can result in U_{PRB} increase from 40% to 62% without rate-limits. This demonstrates that unmanaged FOTA downloads are not recommended in LTE networks with current capacity constraints.

2.2.2 Impact on throughput. Using similar simulation settings as before, we explored the impact of introducing FOTA downloads to throughput of other users in a lightly loaded cell (14% utilization). Figure 3 shows the percentage reduction in average throughput of other users after FOTA downloads were added. The impact varies depending on the rate limit imposed on FOTA traffic. It is not surprising that the largest impact is caused when FOTA is not rate-limited. There is still some impact as rate-limiting is introduced. The result shows that FOTA can significantly reduce throughput of other users, even in lightly loaded cells. Therefore, reducing the load in busy cells is even more advantageous to all users.

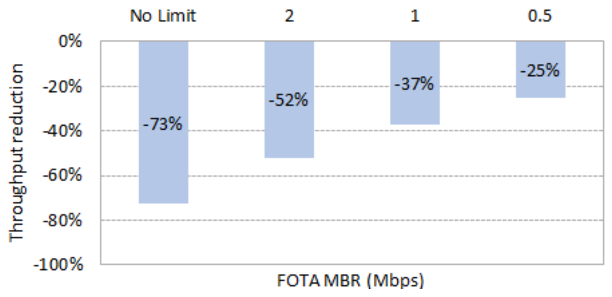


Figure 3: FOTA reduces throughput of other users.

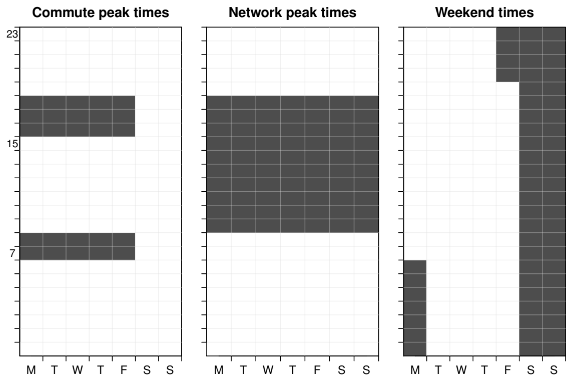
2.3 How to address impact of FOTA

From simulation results, we can infer that uncontrolled FOTA can have significant impact on the cellular network and that rate-limiting helps curtail this impact. There are also several obvious unmanaged approaches to FOTA management, some even naturally come out of our simulation results.For example, we see that since FOTA can harm network performance, even at moderate loads, we need to look at techniques such as offloading FOTA to off-peak hours to alleviate its impact. However, not all approaches may be applicable due to the highly constrained and complex multi-actor nature of this problem. We discuss why this is the case and why we opt to explore a scheduling-based approach.

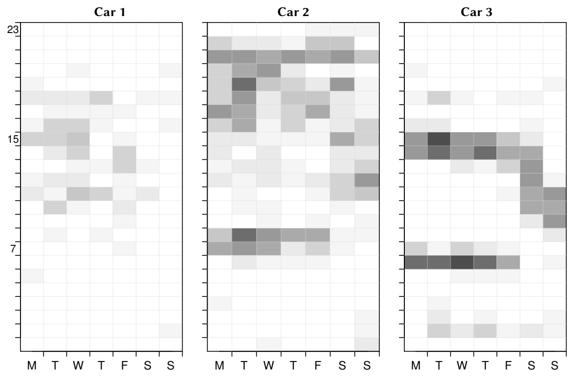
Busy-hour blackouts: To at least avoid sustained overload conditions during network peak time, one solution is to naïvely black out the busiest network hours of the day for FOTA downloads. However, some cars may appear on the network only during busy hours, rendering them unable to receive FOTA under this regimen.

Rate limiting: Eventhough our simulation results show that setting a Maximum Bit Rate (MBR) on FOTA is a good way of minimizing impact on non-FOTA users, there are multiple problems with it. First, the limits should be applied to only the FOTA flows and not other traffic to the car. This involves a level of complexity to differentiate FOTA flows, which may not be feasible in all networks and FOTA management systems. Second, a simple rate limiting scheme will impair the ability of cars on off-peak cells to capitalize on available bandwidth and cause extended campaigns. We study this case in more detail in §3. Third, it is difficult to determine a single MBR that would work to conserve bandwidth in all radio cells at all times and prevent competition with other users during peak hours. Too low MBR leaves the cells underutilized (such as 1 and 0.5 Mbps in Figure 1). Finally, multiple cars downloading FOTA at any MBR may still cause unwanted levels of overload in a cell.

Quality of service Class Identifier (QCI) is a mechanism in LTE to assign priorities to different classes of traffic. There are 9 QCI levels with several more added recently, but there are difficulties in using them for FOTA. A small number of QCI levels dedicated to best-effort traffic (typically QCI 8 or 9) forces the operator to assign FOTA to probably lowest priority (QCI 9) but the exact priority weights required for FOTA may not match other types of traffic at this QCI level. This solution may not work across different operators and requires a tight integration between the operator's LTE implementation and the manufacturer, which may not be desirable at all times or for every manufacturer. We would instead prefer to have a universal solution.



(a)



(b)

Figure 4: (a) Visualization of important time ranges in the week; (b) Usage patterns from 3 sample cars.

3 FOTA SCHEDULING APPROACH

To address the issues with alternatives in Section 2.3, we propose and implement a FOTA scheduling scheme which represents a form of admission control for FOTA traffic. The schedule sets each car's admission time to the system, i.e., from which time it can download whenever it's local conditions allow, such as the engine running or having a cellular signal. Scheduling cars allows us to manage when and where cars get selected for a FOTA update, allows for opportunistic use of available cell capacity, and avoids over-subscription within radio cells. This approach does not require special assignment of priority inside the operators networks and is based on standard operational measurements and logging typically available in cellular networks. We describe the details of this approach in this section.

3.1 Methodology

In our approach, we employ models of network load and resource utilization, as well as cars' usage of the network, based on historical analysis, to schedule cars for admission to download FOTA.

The key ingredients of our *network model* are PRB utilization (U_{PRB}) and data volume (V_{data}). Using periodic measurements from a large operational cellular network in the U.S., we compute the average U_{PRB} and total V_{data} for each 15-minute time-bin for each radio cell. We build a historical model of $R = V_{data}/U_{PRB}$ as an 8-week moving average. Since both U_{PRB} and V_{data} have a highly predictable diurnal pattern, we can compute the model of R for each cell. We use R to compute how much FOTA data to schedule. In typical cellular networks, several cells cover the same geographic area (*sector*) with non-interfering frequency bands. Hence, we aggregate the model per sector by averaging R values over all comprising cells. We therefore refer to sectors when discussing actual scheduling decisions.

In similar fashion, we generate the *model of car usage* of the network by observing the car modem connections to the network infrastructure. Our data, based on Call Detail Records (CDRs), provides information about radio-level connections made by 1 million cars to the cellular network, such as times and duration of connections, as well as cells and base stations that they connect to. This produced a set of nearly 1 billion connections. Using domain knowledge of radio connection management by the cellular network we are able to link radio activity to car activity and reconstruct cars' sessions on the network from the individual connections. This type of monitoring is anonymous; it does not provide any association to drivers or owners of cars, nor visibility into the content transferred. Through knowledge of the cars' locations in the network and times of their sessions we can make broad inference of patterns of usage.

We note that many cars exhibit strong daily and weekly patterns. By aggregating data over time onto a 24×7 matrix we can use it for operations like filtering peak usage times or aggregating counts of occurrences. Note that this is strictly network location analytics in terms of connections to radio cells, not physical location tracking. Figure 4a shows masks for commute and network peak hours as well as weekend times. Figure 4b shows the usage frequency matrices for 3 sample cars. Each hour is represented by a shaded box, with darker colors representing a higher number of car's connections to the cellular network over 8 weeks. A white box means that the car has not connected during that hour over the study period. Note that Car 1 is a rarely seen in the network. Car 2 depicts a strong usage during evening hours, and very busy Mondays and Fridays. Car 3 displays strong commute behavior during early in the morning, and late in the afternoon during weekdays, and predictable behavior in the weekends.

Using their usage pattern, we can segment the cars as follows:

- Rarely-seen cars, which appears once a while in the network;
- Predictable off-peak cars, which appear during off-peak (or non-busy) network hours frequently;
- Predictable busy-hour cars, which connect to the network during its busy times.

3.2 Rarely-seen and off-peak car scheduling

A rarely-seen car has a low probability of being addressed during the FOTA campaign window. To help maximize completion rate of the campaign, we schedule the rarely-seen cars at random times during the first day of the campaign. Thus, if the car comes online at any time during the campaign, there is a chance it will complete. This rationale is most beneficial to the manufacturer and owner's goals, though the network operator also has an incentive to eliminate future work and uncertainty.

Cars that present off-peak behavior should be scheduled at those times, as much as possible. This approach offloads traffic away from peak hours and it enables those cars to complete faster due to the expected higher achievable throughput during off-peak hours. Using peak-times masks and frequency matrices like in Figure 4, we seek off-peak times where the car was observed at some frequency over the study period. This approach generates a list of candidate periods for each car, where they are scheduled. Note that the peak-times mask and other thresholds can be adjusted according to the problem space (specific FOTA campaign target duration, FOTA download size) or the desired effect (maximizing completion rate, minimizing campaign duration).

3.3 Busy-hour car scheduling

Cars with strong behavior during peak/busy hours may impact the network significantly, and therefore, need to be scheduled carefully to avoid negative network impacts described in Section 2. For this purpose, we developed a sophisticated combinatorial optimization algorithm that is able to minimize (in near-optimal sense) the campaign completion time and, at the same time, address the network constraints.

Since this scheduling problem involves hundreds of thousands of cars, exact approaches are not able to deliver a schedule in acceptable time frame. For this reason, we choose to use the Biased Random-Key Genetic Algorithm (BRKGA), which is an optimization framework able to obtain near-optimal solutions in reasonable time [9]. BRKGA has been shown to be useful in a wide variety of combinatorial and machine learning problems [1–3].

BRKGA evolves a population of solutions $\mathcal{P}$ represented by vectors $v \in [0,1]^n$ in a unitary hypercube of dimension n. Therefore, to obtain a solution for the problem, the vectors/chromosomes must be *decoded* using a function $d : [0,1]^n \rightarrow \mathcal{S}$. The decoder guarantees that the BRKGA framework is independent of the structure of the problem (for details, see [10]).

Consider that n is the number of cars, and that they are indexed by $\{1, \ldots, n\}$. Each allele/key v_i of the chromosome $v \in [0,1]^n$ is paired with the car i. Algorithm 1 depicts a decoder that translates a chromosome to a schedule. This procedure is a straightforward constructive algorithm based on the management of the PRB capacity of the sectors and the historical data of the cars and sectors over time. First, we order the cars according the value of their keys. For each car, we try to fit the car into available spots with enough capacity, trying to minimize the completion time.

Note that the car assignment is based on the amount of available PRBs in the sector. Therefore, we must compute the amount of PRBs and volume of data used by the car. This computation is done using historical data: for a given car c, sector s, and period p, we use the average time that car c spent under sector s (t_{csp} in seconds), the overall per-hour throughput of sector s (τ_{sp} in Mpbs), and its ratio $V^{sp}_{data}/U^{sp}_{PRB}$. Therefore, the additional estimated increase in PRB utilization by this car is computed by

$$\text{PRB utilization}_{csp} = \frac{t_{csp}\tau_{sp}U^{sp}_{PRB}}{V^{sp}_{data}}. \tag{1}$$

The PRB computation is done for each assignment and it is determined by the car, the sector, and the period the car appears under the sector. We also have to guarantee that the maximum number of simultaneous connections is not exceeded, if necessary. This condition is added for generality as a global constraint, but could also be expressed in various other ways. It ensures that local optimization decisions do not violate constraints that other network elements might have.

Algorithm 1: Building a schedule – decoder.

```
Input: Chromosome/vector $v \in [0,1]^n$.
Output: Feasible schedule and completion time.
Let $\pi$ to be a permutation of cars induced by the
  non-increasing order of correspondent keys in $v$;
foreach car in a given order $\pi$ do
    (1) Choose the first period when the car appears on
      network. If the car appears in multiple sectors, assign
      the car to the sector it spent most time in, if:
        a) the sector has available capacity;
        b) the maximum simultaneous connections will not be
           exceeded;
        c) the car spends a minimum amount of time under that
           sector;
    (2) Compute the amount of data that the car can
      download from the assignment above;
    (3) If the assignment is not enough to finish the
      download, look for the next period the car show up
      and the respective sectors. If none of these sectors has
      available PRBs, undo previous assignments for this car
      and start the schedule for this car again from the next
      period, i.e, shift forward a period;
return A feasible schedule and average completion time.
```

Algorithm 1 returns the average completion time as a performance metric to be used by BRKGA to classify the solutions. The average completion time is the sum of the (expected) completion times divided by the number of cars. Although this metric may push some cars to delay their downloads, it helps to build a balanced schedule, and is commonly used in practical scheduling problems.

Another natural metric that can be used is the *makespan*, i.e., the time that the last car completes its download, which can define the campaign length. However, there is a caveat when using makespan in FOTA scenarios: some cars may appear very late during the campaign window, and they will "set" the makespan. In such cases, the completion times of previous cars do not impact the makespan and they may be shifted close to the end of the campaign. From the OEM perspective, this can introduce unnecessary delays for other cars that could finish earlier. One could use the percentile of cars that are able to complete the download, as a metric to be optimized. However, as in the makespan case, to optimize the percentile does not account directly for each completion time and may introduce unnecessary delays in cars that are able to complete earlier.

3.3.1 Pre-processing. Before submitting the cars to the busy-hour scheduler, we perform two important pre-processing steps. The first step is the partitioning of the cars based on geographical region. The rationale is straightforward: most cars usually drive within a certain geographical area and do not impact radio cells far away. Second, since a large number of cars creates a very large combinatorial problem, we reduce the size of algorithm inputs significantly. The partitioning results in several dozen regions which are treated as inputs to separate instances of the scheduler.

Note that some cars may cross the boundaries of two or more regions. In such cases, we schedule the cars in both regions separately assuming that the car will be in each region exclusively. This approach is slightly conservative. However, it does not impact the entire schedule since the proportion of such cars is relatively small compared to the whole population.

The second step consists of the analysis and consolidation of vehicle sessions. We build a session matrix for each car consisting of 24×7 periods for each radio that the car hits. Since a car may have session overlaps in a given time of day, we remove such overlapping sessions as follows. First, we sort the sessions in non-decreasing order of timestamps. For each session in this order, we check if that session has an overlap with sessions taken previously. If an overlap exists, the session if discarded. Otherwise, the session is kept.

4 FOTA CAMPAIGN SIMULATION

We simulate the FOTA campaign according to the following process. Data collection and model building covers 8 weeks for network model and 30 days for car usage model prior to the campaign. Scheduling follows after the models are built. The campaign period is then simulated, during which we observe car behavior and simulate the FOTA downloads. We create FOTA schedules for different scenarios defined by scheduling strategy, download size and MBR.

We use download sizes of 128 MB and 512 MB. Note that a 128 MB download might be expected to complete in a single day's activity of a car, whereas 512 MB might span days. We also evaluate impact of MBR at 1 and 4 Mbps (constrained bandwidth-sharing case and impact of MBR), and 50 Mbps (unlimited bandwidth capped only by LTE and device capabilities).

For each case, we used a population of one million cars and a network trace comprising more than 200,000 sectors (does not represent the whole network). Network model contains PRB utilization and data volume, which for 7 days and 15 minute timebins makes 672 data points per cell. These points are averages over 8 weeks. For car usage sessions, we consider 30 day history which gives us around 79 million sessions aggregated from about 1 billion radio-level connections and process this information according to Section 3.3.1. The definition of off-peak we used for this experiment was evenings and weekends local time. The predictability threshold for reliable off-peak cars was 70%. Based on these assumptions and thresholds, the car usage model produces the following segments:

- 10.8% rarely-seen cars,
- 42.8% off-peak predictable cars for heuristic schedule,
- 46.4% busy-hour predictable cars for optimized schedule.

Therefore, the key impact on the network is caused by 46.4% of cars. The busy-hour scheduler creates the schedule and also identifies cars that can not be scheduled since they appear only on busy radios and cause the PRB capacity to exceed if scheduled. The number of such cars varies significantly due to the size of FOTA download, while the influence of MBR is minor. For 128 MB files, an average of 21% of busy-hour cars can not be scheduled. For 512 MB files, the average is higher at 50%. The reason is that 512 MB files require multiple sessions to be downloaded, and it is much harder to find consecutive slots with free capacity for those downloads based on the times the cars spend connected to the network. We look for consecutive periods to fit the entire download, to avoid cars keeping the download going into the next period that may be busy. Recall that once cars are admitted to the system, they will download whenever the engine is running, without the ability to interrupt and resume downloading according to network conditions. These cars are scheduled randomly during the first day of the campaign.

To assess the effectiveness of our approach (which we call FotaSched), we compare its results with a straightforward random schedule strategy: the cars are spread over the campaign duration uniformly at random. We called this strategy Random.

For busy-hour vehicles, the stop criteria for the scheduler are either the maximum of one wall-clock hour, or 1,000 iterations without improvement on the average completion time of the schedule, whichever comes first. Given the example target of 16 days to complete the campaign, we schedule over a shorter period, in this case 10 days, to give cars some buffer to complete.

The schedules produced by Random and FotaSched are submitted to a simulator, which uses network information (PRB utilization, data volume) and car sessions and locations during the simulated campaign. Therefore, the simulation uses the actual vehicle location in the network to evaluate the effectiveness of scheduling, including the additional 6 days past the schedule to cover the target campaign duration. The simulator keeps track when the car is scheduled (assuming it is notified at the same instance), when the download actually starts (startup delay) and when it ends (completion time). In this way we can assess the impact of different scheduling strategies on the campaign and on the network.

4.1 Results

Since the network has no control over the cars, it is important to assess the prediction ability of our strategy. Table 1 shows the performance metrics for both FotaSched and Random. The first three columns show the configuration parameters of each scenario, and the strategy, respectively. *Startup delay* is the median time between scheduled and actual time instances when FOTA downloads started. Lower delay reflects a more effective schedule and allows more opportunities to download.

Incomplete downloads is the total number of cars that were not able to finish their download within the campaign window, and *Data on busy radios* shows the percentage of data downloaded on busy radios for the two scheduling strategies.

The key benefits of FotaSched are clear from the results. The schedules produced by FotaSched have a significantly shorter median startup delay (48% ± 7.69 of the Random). *Therefore, FotaSched can predict the car activity well within the granularity of the scheduling timebins. This capability helps time-shift traffic from peak to off-peak hours, enhancing the overall performance.* Hence, elapsed time from scheduled start to completion is much lower using schedules from FotaSched, given generally the same download duration for the two scheduling strategies. For example, 128 MB download takes about 20 seconds in general. Note that it is expected that the median startup delay for Random is the same across scenarios, as

Table 1: Schedule statistics for the simulation scenarios.

MBR (Mbps)	Size (MB)	Scheduling Strategy	Startup delay (minutes)	Incomplete downloads	Data on busy radios
1	128	RANDOM	397	53K	6.5%
		FOTASCHED	189 (52%)	41K	4.1%
	512	RANDOM	397	108K	7.5%
		FOTASCHED	266 (33%)	97K	7.1%
4	128	RANDOM	397	36K	6.2%
		FOTASCHED	194 (51%)	27K	6.1%
	512	RANDOM	397	108K	6.5%
		FOTASCHED	182 (54%)	44K	4.4%
50	128	RANDOM	397	24K	—
		FOTASCHED	199 (49%)	19K	—
	512	RANDOM	397	30K	—
		FOTASCHED	193 (51%)	24K	—

these schedules are the same (independent of MBR and file size, unlike with FOTASCHED).

Next, *the number of incomplete downloads improves from RANDOM to FOTASCHED,* which is a significant benefit to the OEMs, car owners and somewhat to the network provider. The reduction can range from 10% to 59%, depending on the scenario, with the savings to the OEM amounting to about $1M per 10,000 cars that do not have to receive FOTA at the dealership. Note that cars actually complete the downloads past the campaign window, while our definition of *complete* here means "within the campaign window".

Then, FOTASCHED *reduces the load on busy radios,* with the most noticeable effect for [128 MB, 1 Mbps] and [512 MB, 4 Mbps] cases. These are the cases where the download is far more likely to succeed within one daily cycle, and hence the optimizer finds it much easier to assign available slots for download. This effect is depicted in Figure 5. For one heavy weekday in the campaign, we see that FOTASCHED can hold down [512 MB, 1 Mbps] FOTA to roughly constant load over increasing U_{PRB} decile relative to the linearly increasing RANDOM. In the [512 MB, 4 Mbps] case it achieves diminishing load in the higher U_{PRB} deciles, which is the goal of the optimized schedule, i.e., avoiding the busy cell overload ($U_{PRB} > 80\%$) that reduces performance for everyone. We don't perform this analysis on unlimited MBR as cars complete very quickly.

Finally, Figure 6 depicts the time series of data volume delivered over the campaign for a file size of 128 MB and MBR of 1 Mbps. The vertical lines show the boundaries of the campaign with the shaded areas showing the weekends. Note how FOTASCHED pushes more traffic at the beginning of the campaign and during off-peak weekend compared to RANDOM, while at the same time reduces data on busy radios, as shown in Figure 5 and Table 1.

Therefore, *FOTASCHED exploits otherwise unused radio capacity early in the campaign and during off-peak time, while reducing the average completion time and load to busy cells.*

5 RELATED WORK

Various forms of time-shifting, offloading, scheduling and pre-fetching data have been studied before in the context of cellular networks. When users are directly consuming the data, they are willing to accept time-shifting when offered right incentives [11, 26]. Recent work has shown that applications can almost automatically

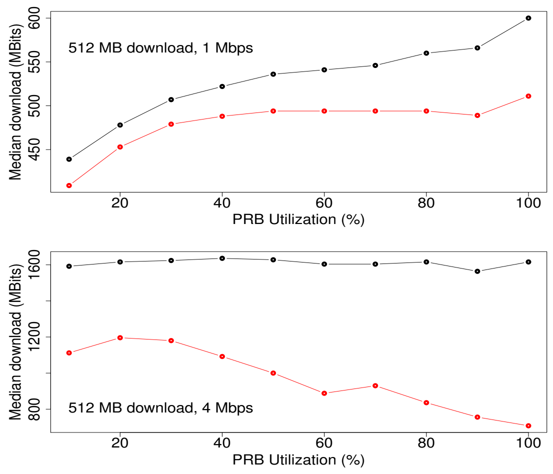


Figure 5: Data allocation to radio by PRB decile (black lines are RANDOM, red lines are FOTASCHED).

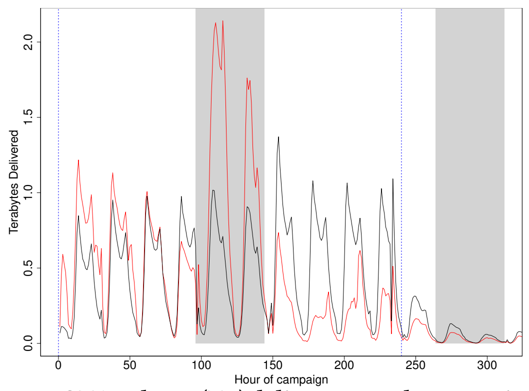


Figure 6: FOTA volume (TB) delivery over the campaign time (black line is RANDOM, red line is FOTASCHED).

be made delay-tolerant [17]. Unfortunately these approaches are not applicable to FOTA.

Prior work on time-shifting has largely focused on predicting network demands and pre-fetching content over time scales of seconds. CoAST [27] demonstrates that peak utilization can be reduced, by scheduling traffic over time scales of seconds to flatten congestion spikes. It has also been demonstrated that traffic shaping can reduce peak congestion, at the cost of some loss in performance [16]. These approaches do not solve the problem for FOTA downloads. Time-shifting on the second scale cannot mitigate the situation where many cars initiate very closely or overlap the downloads, and especially when FOTA downloads would represent a majority of traffic in a cell. It also does not help predict future to-the-minute time scales that FOTA downloads may require. Traffic shaping may work for a small number of FOTA downloads in a cell, but do not help generally because they may not utilize the capacity when available and hence reduce the effectiveness of the campaign.

Other work on short-term time-shifting includes Informed Mobile Prefetching [13], which decreases latency on mobile devices while meeting data and power budgets; Wiffler [4], which focuses on offloading data from 3G to WiFi, delaying traffic by 30-60 seconds, and work by Han et al. [12] which reduces load on cellular networks by sharing popular downloaded traffic in a peer-to-peer manner. Conversely, Procrastinator [23] avoids pre-fetching data when it could potentially lead to excessive data usage.

There is also prior work related to network load forecasting. Breadcrumbs [20] discovers that user-experienced performance can be predicted from the device's perspective 30 seconds in advance. Bartendr [25] predicts signal strength seconds in advance and time-shifts streaming and syncing traffic to better utilize the network over short time periods. We forecast hours and days in advance, and focus on predicting cell capacity and average data throughput directly from network data. Laner et al. [14] demonstrate that there are global network congestion trends, but that individual cell towers may differ substantially from this standard behavior. Xiong et al. [31] show that the next cell tower a user visits can be predicted given the current one.

Targeted solutions to enable communication between vehicles and road-side infrastructure are promising for FOTA, but they are in their infancy; hence they cannot address the current needs of millions of cars on the road that need FOTA now and will need in years to come, but lack support to connect to such infrastructure [30].

While some insights from these efforts are valuable, they cannot directly address the FOTA use case since they do not consider the specific current constraints it imposes, such as limited on-network time or limited FOTA campaign duration. These constraints are far more relaxed for smartphones, since they are continuously connected and have looser regulatory or safety considerations. Some specifications and protocols for remote device management are defined by the Open Mobile Alliance Device Management (OMA DM) [21]. While an OEM can use OMA DM to manage devices, it is limited to protocols for data exchange, data formats, security and fault management. Understanding of how devices behave and when to schedule their particular download is still heavily dependent on the network context and requires cooperation between the OEM and the cellular operator.

6 CONCLUSION

We present the design of the scheduling system for managing large FOTA downloads in a cellular network by combining load, usage and location analytics with the heuristic and optimized scheduler. The system builds network and connected car predictive usage models and applies them to schedule admission of cars to download FOTA. The massive network-scale problem is reduced by segmenting cars and using heuristics to schedule some segments, leaving the most complex segment to the optimization framework, namely the Biased Random-Key Genetic Algorithm.

We show that network-scale analysis involving millions of devices and hundreds of thousands of radio cells is possible and that the FOTA management system built on top of it significantly improves on trivial solutions, even when device and network models are using high levels of aggregation over time. Our system achieves the key design objectives: reduction of network load during busy hours and faster completion of downloads for individual cars. We also reduce uncertainty in network management by minimizing the durations that cars are in the state of being ready and admitted to inflict heavy network load but not yet doing so.

REFERENCES

[1] C.E. Andrade, S. Ahmed, G.L. Nemhauser, and Y. Shao. 2017. A hybrid primal heuristic for finding feasible solutions to mixed integer programs. *European Journal of Operational Research* 263, 1 (2017), 62–71.

[2] Carlos E. Andrade, Mauricio G. C. Resende, Howard J. Karloff, and Flávio K. Miyazawa. 2014. Evolutionary algorithms for overlapping correlation clustering. In *Proc. GECCO*. 405–412. https://doi.org/10.1145/2576768.2598284

[3] Carlos E. Andrade, Mauricio G. C. Resende, Weiyi Zhang, Rakesh K. Sinha, Kenneth C. Reichmann, Robert D. Doverspike, and Flávio K. Miyazawa. 2015. A biased random-key genetic algorithm for wireless backhaul network design. *Applied Soft Computing* 33 (2015), 150–169.

[4] Aruna Balasubramanian, Ratul Mahajan, and Arun Venkataramani. 2010. Augmenting Mobile 3G Using WiFi: Measurement, Design, and Implementation. In *Proc. ACM MobiSys*.

[5] Horatiu Boeriu. 2015. BMW to offer over-the-air updates for BMW navigation maps. (2015). http://goo.gl/wshQQL

[6] ComputerWorld. 2015. Over-the-air software coming soon to your next car. https://goo.gl/Cz45rp. (Feb 2015).

[7] Ericsson. 2016. Ericsson Mobility Report. (2016). https://goo.gl/wGUALK

[8] Ford. 2016. How to download SYNC software updates to a USB drive. https://goo.gl/WpabXj. (2016).

[9] José F. Gonçalves and Jorge R. de Almeida. 2002. A hybrid genetic algorithm for assembly line balancing. *Journal of Heuristics* 8, 6 (2002), 629–642.

[10] José F. Gonçalves and Mauricio G. C. Resende. 2011. Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics* 17 (2011), 487–525. Issue 5.

[11] Sangtae Ha, Soumya Sen, Carlee Joe-Wong, Youngbin Im, and Mung Chiang. 2012. TUBE: Time-dependent Pricing for Mobile Data. In *Proc. ACM SIGCOMM*.

[12] Bo Han, Pan Hui, V.S. Anil Kumar, Madhav V. Marathe, Guanhong Pei, and Aravind Srinivasan. 2010. Cellular Traffic Offloading Through Opportunistic Communications: A Case Study. In *ACM CHANTS*.

[13] Brett Higgins, Jason Flinn, T. J. Giuli, Brian Noble, Christopher Peplin, and David Watson. 2012. Informed Mobile Prefetching. In *Proc. ACM MobiSys*.

[14] Markus Laner, Philipp Svoboda, Stefan Schwarz, and Markus Rupp. 2012. Users in Cells: a Data Traffic Analysis. In *WCNC*.

[15] Kyunghan Lee, Joohyun Lee, Yung Yi, Injong Rhee, and Song Chong. 2012. Mobile Data Offloading: How Much Can WiFi Deliver?. In *Proc. CoNEXT*.

[16] M. Marcon, M. Dischinger, K.P. Gummadi, and A. Vahdat. 2011. The local and global effects of traffic shaping in the internet. In *Proc. COMSNETS*.

[17] YoungGyoun Moon, Donghwi Kim, Younghwan Go, Yeongjin Kim, Yung Yi, Song Chong, , and KyoungSoo Park. 2015. Practicalizing Delay-Tolerant Mobile Apps with Cedos. In *Proc. ACM MobiSys*.

[18] Movimento. 2016. Frost and Sullivan Recognizes Movimento With Enabling Technology Leadership Award. https://goo.gl/WRcwUs. (Mar 2016).

[19] Automotive News. 2016. Over-the-air updates on varied paths. https://goo.gl/xzabL4. (Jan 2016).

[20] Anthony J. Nicholson and Brian D. Noble. 2008. BreadCrumbs: Forecasting Mobile Connectivity. In *Proc. ACM MobiCom*.

[21] OMA 2017. Open Mobile Alliance Device Management. https://goo.gl/kks99S. (2017).

[22] Mark Phelan. 2016. Study: More cars will get automatic software updates. (2016). https://goo.gl/JEQt5x

[23] Lenin Ravindranath, Sharad Agarwal, Jitendra Padhye, and Chris Riederer. 2014. Procrastinator: Pacing Mobile Apps' Usage of the Network. In *Proc. ACM MobiSys*.

[24] Redbend. 2015. Updating Connected Car Software Over-The-Air - Why Wait? http://www.redbend.com/en/resources/white-papers#resource=231. (Aug 2015).

[25] Aaron Schulman, Vishnu Navda, Ramachandran Ramjee, Neil Spring, Pralhad Deshpande, Calvin Grunewald, Kamal Jain, and Venkata N. Padmanabhan. 2010. Bartendr: A Practical Approach to Energy-aware Cellular Data Scheduling. In *Proc. ACM MobiCom*.

[26] Soumya Sen, Carlee Joe-Wong, Sangtae Ha, Jasika Bawa, and Mung Chiang. 2013. When the Price is Right: Enabling Time-dependent Pricing of Broadband Data. In *Proc. SIGCHI*.

[27] Cong Shi, Kaustubh Joshi, Rajesh K. Panta, Mostafa H. Ammar, and Ellen W. Zegura. 2014. CoAST: Collaborative Application-aware Scheduling of Last-mile Cellular Traffic. In *Proc. ACM MobiSys*.

[28] Telefonica. 2014. Connected Car Industry Report 2014. https://goo.gl/CJgRrv. (Jul 2014).

[29] TIME. 2014. Your Car Is About to Get Smarter Than You Are. http://business.time.com/2014/01/07/your-car-is-about-to-get-smarter-than-you-are/. (Jan 2014).

[30] U.S. Department of Transportation. 2017. Vehicle-to-Infrastructure (V2I) Resources. (Aug 2017). https://www.its.dot.gov/v2i/

[31] Haoyi Xiong, Daqing Zhang, Daqiang Zhang, and V. Gauthier. 2012. Predicting Mobile Phone User Locations by Exploiting Collective Behavioral Patterns. In *IEEE UIC/ATC*.

[32] Jeff Zurschmeide. 2016. Your next car will update itself while you sleep, and maybe watch you too. (2016). https://goo.gl/1Wtvv8