Prefetching of mobile devices information: a DNS perspective

Antoine Bernard, Mohammed Laroui, Michel Marot, Sandoche Balakrichenan, Hassine Moungla, Benoit Ampeau, Hossam Afifi, Monique Becker

Prefetching of mobile devices information - a DNS perspective

Antoine Bernard*†, Mohammed Laroui†‡, Michel Marot†, Sandoche Balakrichenan*
Hassine Moungla†‡, Benoit Ampeau*, Hossam Afifi† and Monique Becker†
*Afnic - Email : {antoine.bernard,sandoche.balakrichenan,benoit.ampeau}@afnic.fr
†Samovar, Télécom SudParis, Institut Polytechnique de Paris
Email: {antoine_bernard,mohammed.laroui,michel.marot,hassine.moungla,hossam.afifi,monique.becker}@telecom-sudparis.eu
‡LIPADE, Université Paris Descartes, Université Sorbonne Paris Cité, Paris, France
Email: {mohammed.laroui,hassine.moungla}@parisdescartes.fr

***Abstract*—The development of vehicular technologies and infrastructures leads to development in mobility handling for wireless communications. Improving connectivity establishment and reliability became an issue, especially for vehicles that may move out of antenna coverage during connection establishment. This paper focuses on improving LoRaWAN connectivity for roaming devices by combining a machine learning predictor and DNS prefetching to gather information necessary for connection establishment before the device comes under coverage, thus reducing the overall latency for connection establishment. The paper also relates to other issues by comparing the solution with other approaches and studying antenna occupation.**

Keywords - IoT, V2I, V2X, DNS, Prefetching, Machine Learning Prediction, Traffic Prediction, LoRaWAN, Edge Networks

I. Introduction

The evolution of vehicular network generations led to new challenges in the different communication models such as vehicle-to-vehicle communication or vehicle-to-infrastructure systems. The connected vehicles can provide data caching or task offloading for other vehicles and user devices. Besides, the high mobility of vehicles is a major issue directly related to the communication channel where the stability of the connection enhances the quality of service, particularly latency and delay.

The Domain Name System (DNS) is a key component on the Internet that might incur additional latency and hinder device connectivity. From a user's point of view, access must be provided as smoothly as possible, **without additional cost**, to develop the technology's adoption. From an operator's point of view, an increase in latency might incur **congestion** or **antenna overload** which would decrease the Quality of Service for IoT solutions. And in roaming scenarios, serving all users as soon as possible would decrease the impact from other networks on their own gateways. Thus, reducing the impact from DNS requests when a device is joining becomes a **key connectivity concern**. It is particularly challenging in mobile environments.

Also, the issue behind storing and sharing DNS data, or where to locate a DNS cache, how long to keep information cached and when to access it as an operator, is crucial to improve backend network mechanisms. Prefetching information is a common strategy to reduce latency within networks. Web browsers use this technique to obtain IP addresses for domains within a web page, predicting that the user may click on a link, thus sparing the DNS requests when a user clicks by performing the request beforehand.

This paper analyzes two DNS prefetching strategies in an urban scenario involving mobile vehicles in Roma for which we wish to provide Long-Range Wide Area Network (LoRaWAN) connectivity. We evaluate their impacts on both the user and operator's quality of service. We consider mobile vehicles in Roma for which we wish to provide Long-Range Wide Area Network (LoRaWAN) connectivity. In a LoRaWAN scenario, DNS is involved in the roaming procedure. The proposed solution proposes to use of DNS prefetching to query DNS servers based on device mobility to resolve device-specific information between a gateway and a DNS server. Prefetching can be as simple as requesting that nearby gateways prefetch the information (scenario 2 below) but could also rely on recent mobility models based on Machine Learning (ML) predictions (our scenario 3). This article studies the consequences of **prefetching DNS information on antennas** with regards to device mobility. In particular, we check if the information is prefetched adequately with respect to the actual vehicle location by observing the DNS query success ratio. We also study **antennas occupation** based on mobility scenarios to further understand the possible impact of prefetching on antenna cache filling. Actually, different prefetching strategies lead to higher or lower cache occupations for the antennas.

The presented use case focuses on provisioning DNS connectivity data necessary for the join exchange in a LoRaWAN connection establishment procedure, but this method applies to other DNS data querying. It is even more general since the strategies we propose provision information in antennas and may be used for many other user-centric data.

In the next part, we review the related works. Section III presents our studied scenarios. Section IV presents our results, in which IV-A analyses DNS cache results and IV-B describes antenna occupation within the studied area. Finally, our conclusions are summed up in V.

II. Related Works

A. Improving communications using predictors in Vehicular Networks

This part presents existing works that resolve the communication failure problem caused by mobility in vehicular networks. Some of these solutions exploit machine learning techniques to predict devices movement and improve communication efficiency.

Vinod et al. [1] proposed a mechanism of link life prediction that creates an alternative link before it breaks. They validated the results by studying a set of vehicles on a highway; besides, they used a microscopic or macroscopic (traffic flow, traffic density) approach to generate the vehicles' movements. The proposed algorithm uses the velocity and the location of vehicles to predict the route breaks.

Shelly et al. [2] proposed a statistic method for link lifetime in Vanet networks by using an analytical model. They studied the impact of vehicle density, vehicle mobility, and the transmission range and analyzed the statistics of the communication link. Besides, they studied a case of two vehicles (A) and (B), where V_A, V_B and V_r are velocities of vehicle A, vehicle B and the relative velocities of pair of vehicles respectively.

Work in [3] proposed a link duration prediction via AdaBoost algorithm [4]. The proposed steps consist in aggregating the existing link metrics to generate many predictors; each predictor predicts if the link duration is under or over a threshold with high accuracy using the set of link metrics. In the next step, the algorithm determines the duration of the link using all the knowledge collected from these predictors.

Wang et al. [5] proposed a prediction model called extended link duration prediction (ELDP), which allows the vehicle to estimate the link duration with the other vehicles. Simulations in a city and highway show that the speed of vehicles impacts the link duration prediction in Vanet networks. In this work, a normal distribution needs to be used for vehicles speed.

Das et al. [6] proposed a network formation game called NGOMA algorithm for MAC-level re-transmission. It selects one node from the intermediate node, and in the case of a link failure, the formation game is used to select the relay node to re-transmit a packet from the source node to the destination node. The proposed algorithm reduces the delay and enhances the packet delivery ratio.

Similarly, Bhoi et al. [7] used a data forwarding technique to predict the link failure where a link existence diagram (LED) is generated to know the existing vehicles links. The proposed techniques prove their efficiency in terms of end-to-end delay. Nevertheless, the GPS can't detect obstacles and require huge resources.

Authors in [8] proposed a route prediction in Vanet networks to resolve the problem of the communication link failure; they proposed to use machine learning algorithms for prediction and then studied the efficiency of the proposed solution. Simulation results proved the efficiency of machine learning in route prediction compared to real vehicles mobility.

Each proposed solution improved the QoS in vehicular networks, especially solving the problem behind link failure during communications. Nevertheless, it is difficult to prove the efficiency of these solutions in dense networks with a huge number of vehicles. In addition, the impact of different obstacles is not studied in these works. Some of these solutions exploit machine learning capabilities to predict devices movements. Using artificial intelligence to support and predict device mobility can improve link quality and is more suitable for large-scale vehicular networks.

B. DNS performance, caching and prefetching

DNS prefetching relies on a prediction mechanism; the user could click on the link, so its browser performs the DNS query beforehand for all domains that appear within a web page. This simple prediction mechanism can be applied in any circumstances. [9] analyzed DNS traffic with the increase of IPv6 technologies in web hosting and put it in perspective with network traffic increase in Japan, and offered a prefetching-based solution to increase cache hit rate and reduce response times on web browsers. [10] proposes to study DNS queries in the context of web navigation (DNS over UDP requests) by studying when DNS queries are performed and when the information is needed. Their conclusion regarding prefetching is that no supplementary DNS cost appears due to prefetching. A good tutorial on prefetching and its consequences is provided by the Chromium project [11].

Fetching data using DNS comes with a short delay. [12] studied DNS responses with overall results outlining a 200ms response for 70% of their queries, and 90% of queries are realized within 1s. More recent analyses, such as [13] or [14] outline better results by combining anycast technologies and Content Delivery Networks for DNS. [13] studies responses from top resolvers which answer to 90% of their requests within 100ms. Moreover, [14] provides additional information regarding DNS over TLS (DoT) resolution in which they outline failure rates with responses between 130ms and 230ms from top resolvers. Overall, the time inflation from additional security can be outlined around these values.

DNS over HTTPS (DoH) would add another supplementary cost up to 150ms as outlined by [13] measurements on public resolvers. Overall, sending two complete DNS requests completed with DNSSEC integrity check and secured with DoH would cumulate up to 1.1s of queries done within the first exchanges between the ED and the RG.

Our problem is as such: "Would it be possible to reduce that delay in a mobility context to reduce the impact from DNS querying on channel establishment?"

This is why exploiting DNS to prefetch information is useful; as the information is queried anyway, doing it beforehand if possible reduces the overall latency. Prediction algorithms help us determine where to provide the DNS information. This paper aims to analyze how we could reduce the overhead of DNS querying in mobility solutions for vehicular applications studying various scenarios.

III. Use case and solutions

The LoRaWAN join procedure introduces two DNS queries for channel establishment between gateway and backend (Figure 3). This work provides a few insights on possible solutions based on Machine-Leaning-based mobility predictions and information prefetching from DNS servers.

The usual activation flow detailed on Figure 3 for roaming device consists in :

- an exchange between the End-Device and the serving Network Server associated with a nearby gateway
- a DNS query to identify the Join Server associated with the End-Device
- a DNS query to identify the home Network Server
- an exchange between Join Server and serving Network Server
- a response from serving Network Server to the device

We consider mobility traces from devices moving within the city of Roma; Figure 1 shows part of the studied traces traced as a function of latitude and longitude. Each vehicle is traced with ten points, separated from the next one by a 1-minute delay. We considered all DNS entries to be kept in cache for 5 minutes for these simulations. Note that DNS cache congestion will not be studied here.

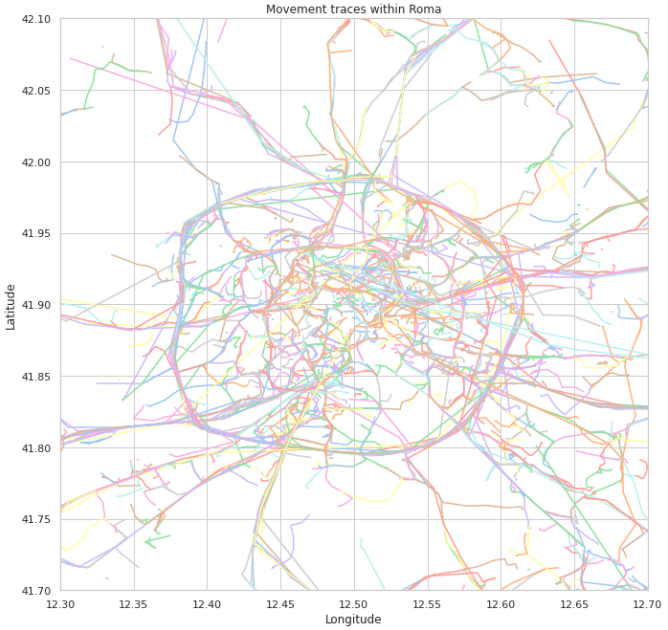

Figure 1: Vehicle mobility around Roma

We simulate antenna placement within the movement perimeter; regularly placed antennas provide independent coverage for our vehicles. Figure 2 shows a vehicular trace with the antenna disposition within its sector. Our test antenna positioning algorithm places the antennas regularly in squares; thus, each antenna has 8 immediate neighbors for all scenarios.

In our simulated fog LoRaWAN deployment, each antenna would act independently and provide access to its devices. To

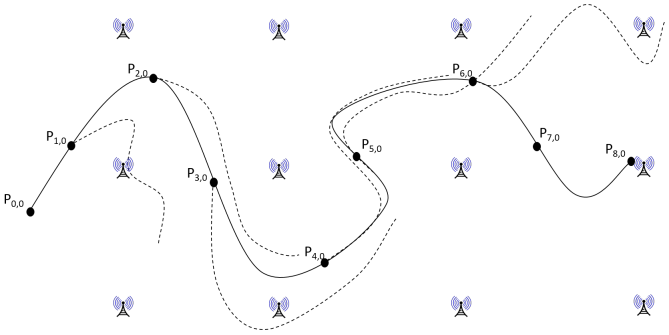

Figure 2: Vehicle and antennas for a single trace

cover a city the size of our perimeter (200 km x 170km), a regular deployment will need around 520 independent antennas to be deployed. With a regular antenna placement and about 8 km between two antennas at most, the vehicle-to-antenna distance will always be bounded between 0 and 4 km.

We assume that each independent antenna will provide roaming access to devices within its reach. As described in figure 3, this means that the antenna will request the device's key from its HN and establish its connection to the ED thanks to them.

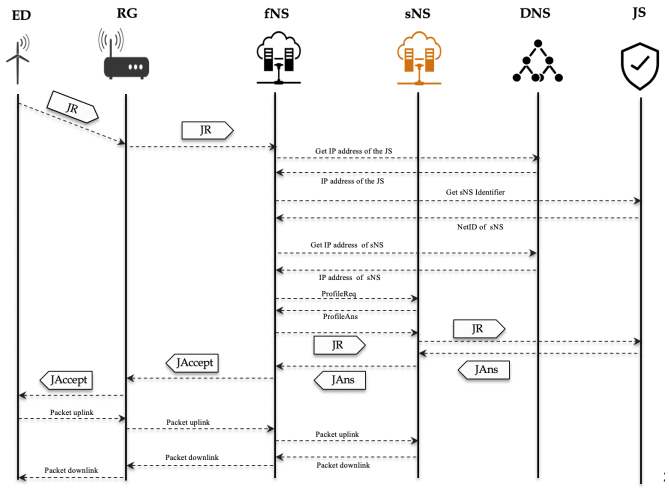

Figure 3: Usual LoRaWAN devices activation message flow

We separated our study into **three scenarios**. In the **first scenario**, **no prefetching** is realized, and the device uses the **standard DNS query** mechanism. It is a reference scenario. In the **second scenario**, we improve the mechanism with a **basic prefetching** mechanism realized by **nearby antennas**: any antenna, in the neighborhood of the antenna under which the device is, is prefetched. Finally, in the **third scenario**, we run **mobility predictions**, using ML, for our devices, plan their **possible future location** and **prefetch the information** based on the predictions.

A. Prediction algorithm

We propose to use a Long Short-Term Memory (LSTM) algorithm to predict vehicles mobility inside the city [15]. The LSTM model is trained using real vehicles mobility dataset [16] in Rome city, Italy. The data represents the real-time vehicles' mobility for one month. The data is classified as follows: vehicle ID (is an integer), date, time, and the position of vehicles (latitude, longitude).

B. First Scenario

For this first scenario, we studied the movements of 6992 devices within the Roma metropolis. Each vehicle is tied to 10 successive locations. We survey the closest antenna for each location and check if the device's information is available on the antenna's cache or should be queried. Actually, depending on the vehicle movements, DNS configuration (number of entries in cache, TTL...) or antenna placement, it may come under an antenna's coverage where it has already been before. The first location of the device is put on the side as "First DNS Query" for consistency with the other scenarios as we would not prefetch information for the first point of the time series.

C. Second Scenario

For the second scenario, we studied the movements of the same 6992 devices; each vehicle is still tied to 10 successive locations. We look for the nearest antenna for each location and check if the device's information is available on the antenna's cache or should be queried.

Our test antenna positioning algorithm places the antennas regularly in squares; thus, each antenna has 8 immediate neighbors. In this scenario, we prefetch the information on these 8 closest antennas to anticipate possible device movements. As mentioned above, the first DNS query for each vehicle is put on the side as "First DNS Query" as these DNS queries cannot be anticipated.

The consequences of DNS prefetching on message flow is described in Figure 4, the information necessary to support the devices' connectivity is recovered before the device's Join Request; thus, the time corresponding to the various queries is saved from the first transmission and realized beforehand.

The activation flow (Figure 3) using prefetching becomes as described in Figure 4:

- a Prefetching DNS query to identify the Join Server associated with the End-Device
- a Prefetching DNS query to identify the home Network Server
- an exchange between the End-Device and the serving Network Server associated with a nearby gateway
- an exchange between Join Server and serving Network Server
- a response from serving Network Server to the device

The first two steps do not need device communication to be processed; knowing that the device is nearby is sufficient.

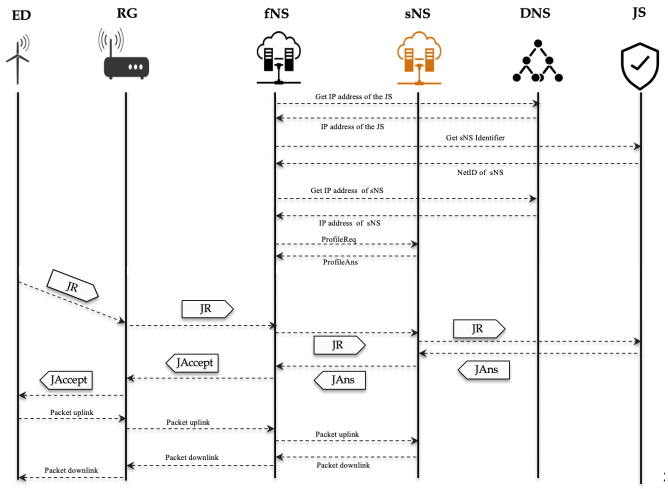

Figure 4: LoRaWAN devices activation message flow with our DNS prefetching mechanism

D. Third Scenario

In this third scenario, we predict car mobility using deep learning algorithms and identify antennas candidate for device coverage. Based on these predictions, the DNS (or its cache) is queried once by the antenna corresponding to the device's position for each given point within the device's movement. Then for the four following predicted positions, the corresponding antenna will perform DNS prefetching as described in Figure 4 to heat its cache for a possible change of coverage from the device.

Figure 5 provides a rundown on interactions between antennas and DNS Servers in the third use case.

	Actual position	T+1 Prediction	T+2 Prediction	T+3 Prediction	T+4 Prediction
Antenna ID (T-5)	Z	L	M	N	O
Antenna ID (T-4)	Y	I	J	K	E
Antenna ID (T-3)	X	G	H	D	S
Antenna ID (T-2)	W	F	C	Q	T
Antenna ID (T-1)	V	B	P	R	U
Antenna ID (T)	A				

Figure 5: Possible solicited antennas in Scenario 3

For a given position A, we consider the 25 possible antennas (B to Z) from the previous predictions and actual positions of the vehicle:

- Antennas B to E are antennas corresponding to the prediction of position A in previous moments in time $\{f_{T-i}(T+i), i \in [[1,4]]\}$). If antenna A corresponds to one of these antennas, we consider that our prediction is

successful, and we hit the cache as the information was prefetched in previous moments in time.

- Antennas F to O correspond to the predictions for previous positions of the device ($\{f_{T-i}(T+j), i \in [[1,5]], j \in [[1,4]], i-j>0\}$). If antenna A corresponds to one of these antennas, but not antennas B to E, our prediction was a failure, but the actual corresponding prediction was correct with a time shift. Furthermore, the prefetching for these antennas was realized; thus, the information is still present in the cache, and despite the prediction failure for this exact timestamp, we hit the cache as the information was not purged yet.
- Antennas P to S are a similar case ($\{f_{T-i}(T+j), i \in [[1,5]], j \in [[1,4]], i-j<0\}$), our prediction was a failure, but the predicted antennas were correct considering a time shift (and would probably be correct for future device positions). Furthermore, the prefetching for these antennas was realized; thus, the information is present in the cache, and despite the prediction failure for this exact timestamp, we hit the cache as the information was not purged yet.
- Finally, antennas V to Z are the actual antennas solicited for the device in previous moments in time ($\{f_{T-i}(T), i \in [[1,5]]\}$). Should all other predictions fail but antenna A correspond to any antennas from V to Z, the prediction is a failure, and so is the prefetching, but the information corresponding to these antennas is still present in the DNS cache from previous requests, we labelled this result "DNS Cache".
- In the case where antenna A ($\{f_T(T)\}$) does not correspond to any antenna between B and Z, prefetching was a failure, and a new antenna was solicited; thus, it must realize a DNS request (labelled "DNS Query")
- Additionally, we separated from these DNS queries the DNS query for the first device's location as antenna B to Z constitute an empty set for this given location.

The activation flow using prefetching is the same as scenario 2 (Figure 4). In this scenario, the first two steps for device activation do not need device communication to be processed; knowing that the device will pass under the antenna coverage in the future is sufficient.

IV. Results

A. Cache hit analysis

Figure 6 presents our global results for all three scenarios.

The "No prefetching case" describes our results for the first scenario. For the 10 locations of our 6992 vehicles, the antenna either queries the DNS as part of the vehicle's first localization, queries the DNS as part of an antenna change for the device or queries its own cache since the device was already known.

Our studied traces are not heavily mobile for now as we study an urban scenario, and additional studies would be necessary to study possible other equilibriums between DNS caching and DNS querying for mobile devices. That explains that our first insight into these results would be that devices

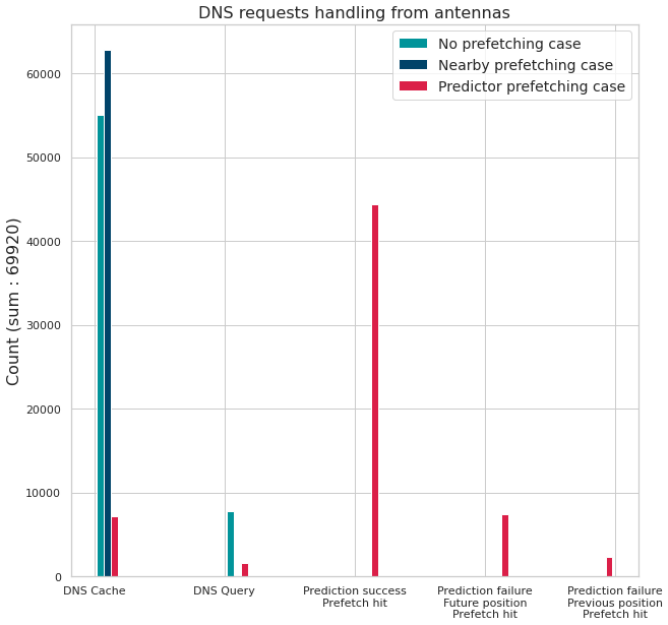

Figure 6: Cache Hit Rate repartition between queries - All cases

are moderately mobile, switching antennas once within the 10 points of their movements, moving around 35km per hour. We observe the 6992 initial DNS requests and around 8 thousand additional DNS queries, consistent with a 2.1 mean antenna per vehicle. The remaining DNS queries are prevented as the request hits the DNS cache within the antenna.

The "Nearby prefetching case" data from Figure 6 describes our results for Scenario 2. As above, for the 10 locations of our 6992 vehicles, the antenna either queries the DNS as part of the vehicle's first localization, queries the DNS as part of an antenna change for the device or queries its own cache as the device was already known through low mobility or prefetching. The simulations show that prefetching permits us to prevent on-the-fly DNS querying. The DNS is still queried but when the information is not yet necessary. The DNS cache handles all queries necessary for device communication, preventing additional DNS query times during handshakes. Nearby prefetching permits us to attain an important hit rate on our cache, whether filled by our first classic DNS query, DNS refresh or prefetched DNS query. A similar situation would be as described in the introduction of section II-B, where prefetching every DNS zone encountered within web page URLs allows to quicken the load time by pre-filling the DNS cache with prefetched DNS queries.

Finally, the "Predictor prefetching case" from Figure 6 combines the results from our vehicle location based on the scenario breakdown from Figure 5. Results are, satisfying compared to the first scenario. Putting aside the first DNS queries, successful prediction leads to hitting an antenna linked to a correctly predicted position in 70.4% of cases. Cache hit rate linked to predictions, whether correct predictions or incorrect predictions by lateness or earliness, would add up

to 86% of requests. The remaining 14% are divided between DNS cache after prediction error (11.4%) and actual DNS queries (2.5%).

B. Antenna occupation

Another important subject to study is antenna occupation. As part of our study, antennas prefetch information based on the possibility that the associated device will pass under its coverage:

- We placed **520** virtual antennas around the city
- Out of them, the first scenario activates **301** antennas. That means that our 6992 vehicles pass near these 301 antennas and that 301 is our minimum number of active antennas as a whole.
- The second scenario activates as whole **393** antennas, a bit over twice more antennas than in the first scenario. The 'nearby case' shows excellent results but would probably create congestion within the network should these results be confirmed at a larger scale.
- Finally, the third scenario activates **380** antennas, globally around the same amount as the antennas solicited as part of the second scenario, Figure 7 gives us more insight on the distribution of these antennas.

Obviously, the absolute number of activated antennas depends on the spatial distribution of the vehicles. Another important criteria is the number of activated antennas *per vehicle*. Figure 7 shows the comparison of the number of activated antennas per vehicle for all scenarios.

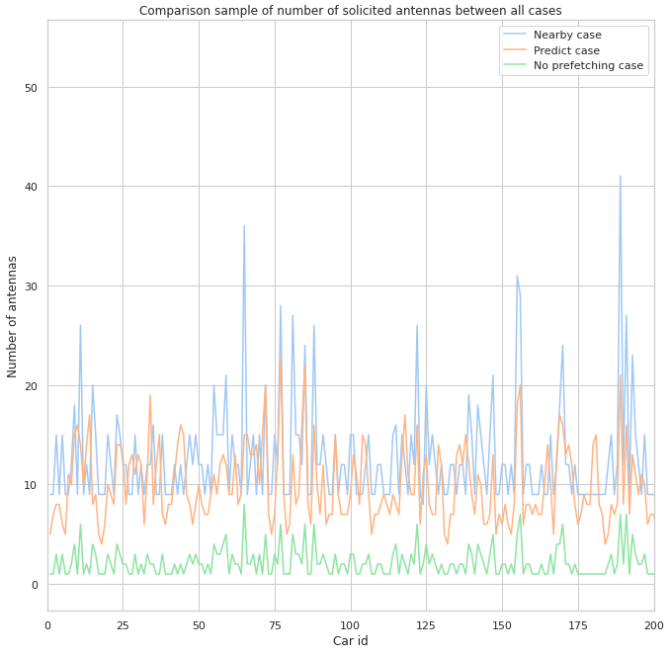

Figure 7: Sample from activated antennas in all scenarios for each vehicle

The mean amount of antennas, as described in Figure 8 activated is as follows:

- Scenario 1 leads to activating 2.1 antennas per moving vehicle on average.
- Scenario 2 leads to activating 12.3 antennas per moving vehicle on average.
- Scenario 3 leads to activating 9.7 antennas per moving vehicle on average.

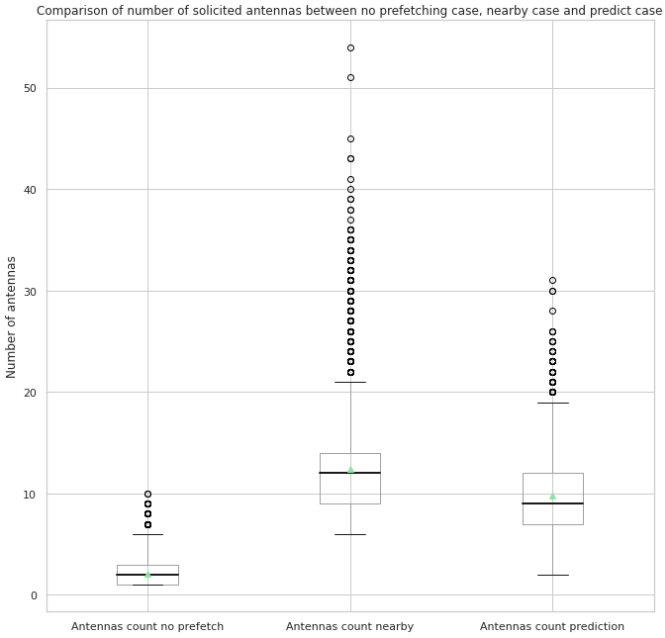

Figure 8: Activated antennas distribution for each scenario

Figure 8 provides additional insight on these values, Scenario 1 has at least 50% of its values between 1 and 3, Scenario 2 between 9 and 14 and Scenario 3 between 7 and 12. This result fits with the moderate mobility from our values as devices that move within 3 antennas would activate around 15 antennas through their movement in Scenario 2. The predictor performs better than the simple nearby prefetching, with more than 75% of its values under the median for Scenario 2. Also, Scenario 2 has many outliers with over 21 antennas solicited per device on highly mobile roads, in which the predictor performs best.

V. Conclusion

DNS prefetching is an **efficient tool** to **reduce the delay added by on-the-fly DNS queries** necessary for device communication. It is a way to prepare the information for the moment the vehicle will be under the umbrella of the right antenna. Prefetching the information on nearby antennas, like in our nearby-case scenario, can completely prevent DNS queries by performing them in advance around the closest ones, but at a cost as more antennas are prefetched than with our Machine Learning (ML) scenario, especially in a highly mobile environment. By exploiting recent ML capabilities for traffic prediction, like in our ML scenario, it becomes possible to **heat the cache** for 86% of requests, and leads to a cache hit for 97% of them, on-the-fly DNS queries following prediction failures constituting 3% of queries.

As described at the beginning of the article, reducing on-the-fly DNS queries leads to time savings. DNS query time is well documented ([9] [10] [12] [13] [14]) and its usual cumulative introduced latency amounts between 60 ms and 250 ms; thus the queries can lead up to a full second of latency saving.

Overall, the ML system would outperform its nearby-activation counterpart in antenna solicitation since only the most likely future gateways are provisioned: scenario 2 activates around 27% more antennas than scenario 3. Also, additional simulations with different antenna location patterns would help to improve this score.

Another interesting further study would be observing antennas overload, such as the overload from DNS cache for which considering a usual 5-minutes caching timeframe. The actual implementation keeps 4096 entries within the cache, each with a variable duration. It would be interesting to study cache congestion against the number of vehicles within the perimeter, considering this 4096 entry limit and the caching duration. Also, considering that our traces amount for taxis which represent around 1% of actual cars circulating around a country, it would be feasible to study the actual overload within the network by increasing the number of vehicles by a 100-factor.

VI. Acknowledgement

This work is a contribution to the Energy4Climate Interdisciplinary Center (E4C) of IP Paris and Ecole des Ponts ParisTech. It was supported by 3rd Programme d'Investissements d'Avenir [ANR-18-EUR-0006-02]. This work was partly financed by the French National Research Agency through the CIFRE program [2018/0668].

References

[1] Vinod Namboodiri and Lixin Gao. Prediction Based Routing for Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, 56:2332–2345, July 2007.

[2] Siddharth Shelly and AV Babu. Probability distribution of link life time in vehicular ad hoc networks. *IEEE International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1418–1423, 2013.

[3] Jun Zhang, Meng Ying Ren, and Houda Labiod. Performance evaluation of link metrics in vehicle networks: A study from the cologne case. *ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications*, pages 123–130, 2016.

[4] Cao Ying, Miao Qi-Guang, Liu Jia-Chen, and Gao Lin. Advance and prospects of AdaBoost algorithm. *Acta Automatica Sinica*, 39(6):745–758, 2013.

[5] Xiufeng Wang, Chunmeng Wang, Gang Cui, Qing Yang, and Xuehai Zhang. ELDP: extended link duration prediction model for vehicular networks. *International Journal of Distributed Sensor Networks*, 12(4):5767569, 2016.

[6] Bhaskar Das and Jalal Almhana. A new cooperative communication algorithm for improving connectivity in the event of network failure in VANETs. *Computer Networks*, 128:51–62, 2017.

[7] Sourav Kumar Bhoi, Munesh Singh, and Pabitra Mohan Khilar. Predicting Link Failure in Vehicular Communication System Using Link Existence Diagram (LED). In *Progress in Intelligent Computing Techniques: Theory, Practice, and Applications*, pages 501–506. Springer, 2018.

[8] Mohammed Laroui, Akrem Sellami, Boubakr Nour, Hassine Moungla, Hossam Afifi, and Sofiane B Hacene. Driving path stability in VANETs. In *2018 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2018.

[9] Kazunori Fujiwara, Akira Sato, and Kenichi Yoshida. Dns traffic analysis — cdn and the world ipv6 launch. *Journal of Information Processing*, 21(3):517–526, 2013.

[10] Mark Allman. Putting dns in context. In *Proceedings of the ACM Internet Measurement Conference*, IMC '20, page 309–316. Association for Computing Machinery, Oct 2020.

[11] DNS Prefetching - The Chromium Projects. http://www.chromium.org/developers/design-documents/dns-prefetching.

[12] Jaeyeon Jung, Emil Sit, Hari Balakrishnan, and Robert Morris. Dns performance and the effectiveness of caching. page 14, 11 2001.

[13] Austin Hounsel, Kevin Borgolte, Paul Schmitt, Jordan Holland, and Nick Feamster. Comparing the effects of dns, dot, and doh on web performance. In *Proceedings of The Web Conference 2020*, WWW '20, page 562–572. Association for Computing Machinery, 4 2020.

[14] Trinh Viet Doan, Irina Tsareva, and Vaibhav Bajpai. *Measuring DNS over TLS from the Edge: Adoption, Reliability, and Response Times*, page 192–209. Lecture Notes in Computer Science.

[15] Mohammed Laroui, Aicha Dridi, Hossam Afifi, Hassine Moungla, Michel Marot, and Moussa Ali Cherif. Energy management for electric vehicles in smart cities: a deep learning approach. pages 2080–2085. IEEE International Wireless Communications & Mobile Computing Conference (IWCMC), 2019.

[16] Lorenzo Bracciale, Marco Bonola, Pierpaolo Loreti, Giuseppe Bianchi, Raul Amici, and Antonello Rabuffi. CRAWDAD dataset roma/-taxi (v. 2014-07-17). Downloaded from https://crawdad.org/roma/taxi/20140717/taxicabs, July 2014. traceset: taxicabs.