Smart Vehicular Routing Based on Crowdsensed Data Using Dynamic Traffic Light Control

Andreea Şonea*, Ciprian Dobre*, Radu-Ioan Ciobanu* and George Suciu†
*Faculty of Automatic Control and Computers, University Politehnica of Bucharest, Romania
Emails: andreea.sonea@stud.acs.upb.ro, {ciprian.dobre, radu.ciobanu}@cs.pub.ro
†Beia Consult International, Bucharest, Romania
Email: george@beia.ro

***Abstract*—The accelerated growth in number of vehicles in recent years has led to an increase in traffic congestion which ends up affecting the quality of life of the citizens. Because of this, emphasis has been placed lately on creating smart cities, where traffic is controlled with the help of technology. Thus, in this paper we address the issue of dynamic traffic control, by proposing a solution where vehicles collect data and forward it to the traffic lights, which can communicate with each other and with the vehicles around them in order to dynamically change their parameters based on the traffic flow, with the final goal of reducing the time spent in traffic and implicitly the overall congestion in the urban area. We then implement and test our dynamic algorithm in the Sim^{2}Car simulator, showing that it is indeed able to reduce congestion in multiple scenarios.**

***Index Terms*—VANET, traffic, dynamic, vehicular, routing, crowdsensing**

I. Introduction

In recent years, traffic congestion has become a critical problem for many developing cities in the world [1]. The situation will only worsen in time [2], becoming a real threat on the quality of life. The main effect is the reduction in vehicle speed, which results in higher commute times, larger fuel consumption and increased pollution levels when compared to fluid traffic. The reason for this phenomenon is that the population is growing and people tend to prefer the benefits of owning a car, such as easy personal mobility and a feeling of security. However, this leads to problems in urban areas where the existing infrastructure was not developed with such large traffic levels in mind, or where methods of alleviating the traffic congestion do not function properly.

The current tendency is to transform existing cities into durable systems known as "smart cities", which are urban areas with various electronic sensors employed for collecting data that help utilize resources in an efficient fashion. This process includes collecting data from the citizens and the devices through crowdsensing, which are then processed and analyzed in order to monitor and manage the transportation, energy, waste management, and information systems, among others. The concept of a smart city integrates ICT (Information and Communications Technology) and a plethora of interconnected devices in order to optimize the services in the urban area. In these conditions, the need for intelligent traffic systems (ITS) arises. An ITS assumes interaction between its components: vehicle drivers, traffic management systems, pedestrians, etc. Some examples of ITS-based improvements are dynamically-timed traffic lights, route planning applications, self-driving cars, etc. Increasing the capacity of the roads is not necessarily the best solution, since it can lead to only temporary positive effects, but to an increase in traffic in the long term [3].

Thus, in this paper we address the issue of dynamic traffic control, by proposing a solution where traffic lights can communicate with each other and with the vehicles around them (through close-range protocols such as Bluetooth, Wi-Fi Direct or NB-IoT) in order to dynamically change their parameters based on the traffic flow. The goal of this system is to respond rapidly to changes in traffic conditions by modifying the durations of the lights and by rerouting traffic participants in order to avoid congestion. In order to test our solution, we employ a realistic vehicular ad-hoc network (VANET) simulator entitled Sim^{2}Car, which uses real-life urban traffic data, where vehicle routes are extracted and replayed, and even changed dynamically [4].

The rest of the paper is structured as follows. In Section II, we present related work in the area of dynamic traffic light solutions. Then, in Section III we describe the scenario where we aim to deploy our proposed solution. Section IV presents our dynamic traffic light switching solution, while in Section V we show and analyze the results obtained when running our algorithm in Sim^{2}Car. Finally, we draw conclusions and present future work in Section VI.

II. Related Work

Due to the increase in congestion mentioned in the previous section, several dynamic traffic control solutions have been proposed recently. In [5], an intelligent traffic control solution that uses wireless sensors for traffic surveillance is presented, which has minimal implementation costs and very good results (decreasing the average waiting time with up to 55% when compared to predetermined traffic). The authors propose an adaptive algorithm entitled TAPIOCA, which employs traffic data to decide the duration of the green lights at an intersection. The system is composed of wireless sensors (built

The work presented in this paper has been funded by the Tel-MonAer project, subsidiary contract no.1223/22.01.2018, from the NETIO project with ID P_40 270 and MySmis Code 105976. This research is also supported by projects CRESCDI ("Susţinerea creşterii capacităţii instituţionale de cercetare a Universităţii Politehnica Bucureşti", contract no. 25PFE/2018) and SPERO (PN-III-P2-2.1-SOL-2016-03-0046, 3Sol/2017).

to ensure the communication infrastructure for traffic) and the controller where the dynamic flow control algorithm runs. The algorithm computes a score for each queue of waiting vehicles based on the number of vehicles on the street and the time elapsed since the last time the green light was set. In order to control multiple intersections, a global score is computed based on the local (per-intersection) scores, the capacity of the roads, and the score defined by the vehicles heading towards the area of the intersections. One drawback of this solution is that it uses user-defined weights for the local score, which can be improved by dynamically computing them based on traffic characteristics (e.g., number of vehicles, route duration, etc.). Moreover, when utilizing dynamic traffic lights, communication can only be performed between directly-connected intersections, thus limiting their synchronization.

Another dynamic traffic lights solution uses fuzzy logic based on the number of vehicles that pass in an intersection at every minute [6]. The proposed method only takes into account the number of vehicles, which can lead to long waiting times at an intersection if the waiting queue is not large. One problem exhibited by this solution is the complexity of the rules when multiple inputs are taken into consideration (and not just the number of cars waiting at the red light). Furthermore, the solution only handles independent or neighbor intersections, thus not having an overview of the entire infrastructure. An improved fuzzy logic-based algorithm is proposed by Collotta et al. [7], whereas another interesting recent solution performs pheromone-based traffic management [8].

Based on the solutions presented above, our proposed dynamic traffic algorithm addresses multiple key issues that are handled partially or not at all in the current literature. These include dealing with cases where vehicles may end up waiting forever at a red light, choosing phases that can have the green light simultaneously so that vehicles can pass safely, or synchronizing the intersections. Our solution automatically computes the traffic lights set to green and their durations, by using not only the length of the waiting queue, but also traffic data such as movement speed, the duration required by a vehicle to cross the intersection, the time elapsed since the last green light, etc. Furthermore, the solution is evaluated using Sim^{2}Car on real infrastructures as a basis for communication, the evaluation being performed for an entire city, not just for a few key connected intersections.

III. Collecting and Validating Traffic Data

The main idea of the solution proposed in this paper is that the dynamic traffic lights have access to information about the road segments that they connect, in particular in terms of the number of waiting vehicles and their behavior. Based on this information, we show in Section IV that the traffic lights are able to make informed decisions regarding the durations and occurrences of green lights, with the purpose of improving the driver experience (in terms of route duration, fuel consumption, gas emissions, etc.). However, an important component of our solution is the way data are collected and aggregated, and how they reach the traffic lights.

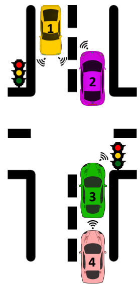


Fig. 1. Vehicle-to-vehicle data collection.

A. Data Collection and Aggregation

In our proposal, we assume that data are collected from the vehicles themselves, using the drivers' smartphones, which are equipped with GPS and a network connection. Most of the previous proposals for crowdsensing in transportation, as well as existing applications (such as Waze[1]), send the data that they collect to a cloud backend, where the aggregation and processing are performed. However, while this is affordable for a very large company such as Google, having an infrastructure that can cover an entire city can be very expensive. For this reason, in our paper we propose employing vehicle-to-vehicle communication for collecting and aggregating data locally, using opportunistic networking-based solutions.

A scenario for vehicle-to-vehicle data collection is presented in Figure 1. In the figure, it can be observed that two vehicles located on the same lane close to the same intersection (numbered 3 and 4 in the picture) are able to communicate with each other through close-range protocols (such as Wi-Fi Direct or Bluetooth), so that they both know that they are advancing in the same direction and waiting at the same traffic light. In this situation, they both have a similar view of their surrounding area. However, since vehicle 3 is in range of the traffic light (which is also equipped with wireless communication), the two can exchange information as well. This way, the traffic light knows that there are two vehicles waiting for the green light, so it can adjust its lights dynamically. Had there been additional cars behind vehicle 4, they would have communicated with each other in a hop-by-hop fashion through close-range protocols.

There can also be situations where vehicles going in opposite directions can communicate, so that they have a better view of the area. This is the case in Figure 1 for vehicles 1 and 2, which meet for a brief period of time while passing each other. However, that time is enough for them to exchange information that might help the traffic lights improve the circulation. For example, vehicle 2 might inform vehicle 1 that there are two cars waiting for the green light at the other end of the intersection, or even notify it about other cars on vehicle 1's direction that vehicle 2 has encountered recently. At the same time, vehicle 1 can transmit these data to the semaphore it is in range of, which can then make informed decisions

[1]https://www.waze.com.

regarding when to change lights. For a better coordination, the two semaphores in Figure 1 can be connected with each other (through a backend or directly), which would allow them to synchronize their lights and further improve the overall performance.

For the communication between the various types of nodes (vehicles, traffic lights, perhaps even a lightweight cloud backend), a fog-based model would be the suitable choice, because, in our scenario, the vehicles are the mobile nodes and the traffic lights are the more powerful fog devices. One such model, which also includes device-to-device communication at the bottom layer, is Drop Computing [9], which takes advantage of the social connections between nodes in order to optimize hit rate, latency and delivery cost.

B. Data Validation

When vehicles exchange data and then the traffic lights use them to make their decisions, it is important that the information used is correct. There may be situations where vehicles send incorrect information due to a malfunction of the smartphone collecting the data, or there may even be malicious nodes that intentionally inject incorrect information into the network. These situations need to be avoided by employing suitable trust and reputation solutions that are able to isolate malicious nodes, while also preventing defective ones from negatively influencing the behavior in the network. One such example is SAROS [10], which takes advantage of the default behavior of an opportunistic device-to-device network by employing gossiping and quorum methods for selecting the correct messages.

IV. Dynamic Traffic Light Control

In this section, we propose our solution for dynamic traffic light control in an intersection, after presenting the simulator used for implementing our solution (entitled Sim^{2}Car).

A. Sim^{2}Car

The advantage of simulators over theoretical mobility models is that they offer a realistic solution of analyzing mobility by utilizing a real-life environment. The analysis of vehicle mobility can be modeled microscopically and macroscopically, through models that are tested and improved using simulators, prior to real-life implementation. The macroscopic models perform an analysis of the traffic as a whole, observing larger components such as the vehicle model. On the other hand, microscopic models analyze each vehicle individually, offering more realistic traffic information. For this reason, the focus in recent years has been on microscopic models [11] such as SUMO [12], VanetMobiSim [13], MATSim [14], or VNSim [15].

The simulator that we devised and employed for implementing our solution, Sim^{2}Car [4], simulates vehicle movement using data collected from GPS devices and street graphs from OpenStreetMap (OSM)[2]. It is an easily extensible application implemented in Java that allows vehicles to dynamically change their movement patterns through mechanisms such as speed adaption, traffic rules, driver behavior, etc. Its main characteristics are modularity, extensibility and response speed.

B. The Architecture of an Intersection

In order to propose a solution for dynamic traffic control in an intersection, we need an intersection model. We thus implemented in Sim^{2}Car a 4-street cross intersection model. Prior to reaching the intersection, each vehicle needs to choose a direction, from forward, left or right. Based on the current position of the vehicle and the movement direction, the following movements can be defined: south-north, south-east, south-west, north-south, north-east, north-west, east-west, east-north, east-south, west-east, west-south, and west-north.

One of the goals of this paper is to implement a solution for dynamic traffic control using real-time data regarding the flow of vehicles in an intersection, with the purpose of reducing the traffic light waiting times. The current traffic state is observed in an intersection, and this information is used to choose a functioning pattern for the intersection traffic lights. This leads to a configuration containing the color of each traffic light (green or red) and durations for them.

C. Dynamic Traffic Control in Intersections

Traffic in an intersection is controlled with the help of traffic lights, which work in phases. Each phase guides the traffic based on a certain movement type, and has a certain period where the green light is set so that vehicles with that movement type can pass. Each street has a traffic light for controlling the access of vehicles in an intersection. For the four types of movement, four phases are defined. One functioning cycle of the intersection contains a succession of these four phases where, for a certain period of time, some lights are green and allow vehicles to safely enter the intersection. If a certain type of movement does not have any vehicles in the waiting queue, it will not receive the green light. Instead of setting fixed durations for the traffic lights, our solution dynamically sets the lights based on vehicle movement and waiting queue sizes. The main goal is to reduce traffic light waiting time and to eliminate cases where vehicles may end up waiting forever. The algorithm allows different types of movements to be executed simultaneously and safely for the drivers.

During the simulation, the intersections are controlled by dynamic traffic lights. This means that the functionality of each traffic light depends on the queues of waiting cars. Each intersection is controlled by a master traffic light which sets and communicates the color and the period of each traffic light it manages. At each step of the simulation, the master traffic light receives information from the cars that are waiting for the green light. After the information is received, the master traffic light decides if the color should be changed or not.

The communication between the cars and the master traffic lights is done via Wi-Fi Direct or Bluetooth. When a car has just stopped at a red traffic light, it sends a message to the traffic light master (either directly or through opportunistic hop-to-hop communication). The traffic light master adds

[2]www.openstreetmap.org/.

the car to the corresponding traffic light queue. During the simulation, the master traffic light decides if the color of the traffic lights should be changed according to the waiting queue length, according to these steps:

1) **Determining traffic volume (number of cars waiting on each queue).** This step assumes that each smart traffic light has knowledge regarding the traffic it controls and the queue of waiting vehicles (this data is obtained by monitoring through crowdsensing).
2) **Choosing the most favorable traffic light (queue) for setting the green color.** The direction that has the highest waiting queue is most likely to see the green light. When setting the green light, the time that has passed since the last light change is also taken into account. If the waiting queue is too long, the time assigned to the green traffic light may be too high, and the other cars waiting to move in other directions can wait too long. To avoid waiting indefinitely, it is not permitted for a queue to wait longer than 120 seconds. Setting the green light for a direction also implies allowing the passage of vehicles that can enter the intersection safely from other directions as well (so one green light might actually lead to the setting of multiple green lights in the intersection). When dealing with multiple intersections, the synchronization of the traffic lights is considered at this step, because creating "waves" of green lights is an important part of decongesting the traffic. Communication between traffic lights in different intersections is performed through a cloud backend.
3) **Choosing the duration for the green light.** This step determines the running time of the traffic lights for the green color just assigned. It should be lower than $T_{max} = 90$ seconds and greater than $T_{min} = 7$ seconds[3].

The solution for dynamic traffic control first needs to define the waiting queues for each traffic light with source s and direction d, where S, W, N and E are the sources (the cardinal points), and L, R and F are the directions (left, right, forward)[4]:

$$Q(s, d), \forall(s, d), s \in \{S, W, N, E\}, d \in \{L, R, F\}$$

Then, the waiting time for the first vehicle at the traffic light is defined as the duration between the moment when a vehicle reaches a traffic light and the current time:

$$T_{first}(s, d), \forall(s, d), s \in \{S, W, N, E\}, d \in \{L, R, F\}$$

In order to eliminate the case where vehicles end up waiting forever at the traffic light, the following logic is followed. If, for one or multiple traffic lights, the vehicle in front has a longer waiting time than the maximum defined waiting time ($T_{wait} = 120$ seconds), then the vehicle that has waited the longest gets the green light in the following phase[5] (G is the set of green lights):

$$\exists(s', d'), T_{first}(s', d') \geq T_{wait} \rightarrow G = G \cup (s', d')$$

The next step is to compute the length of the vehicle queue that gets the green light. If the longest waiting queue has been selected at the previous step (by waiting more than the maximum threshold T_{wait}), then the green light duration is computed by selecting the queue with the highest size out of all the queues whose vehicles can advance safely in the intersection with the previously chosen movement direction:

$$Q_{phase} = max(Q(s', d'), Q(s, d)), \forall(s, d)$$

If no vehicle waited more than T_{wait}, the queue length is computed as follows:

$$Q_{phase} = max(Q(s, d)), \forall(s, d)$$

At the next step, we compute the time it takes for all the waiting vehicles in the selected queue to cross the intersection (where T_1 is the time needed for the first vehicle to cross):

$$TPQ_{phase} = T_1 + \alpha \times (Q_{phase} - 1)$$

Having computed this, we compute the duration of the green light for the previously selected queue and direction. In this case, in order to avoid infinite waiting times in certain situations, we limit the duration of the green light to a maximum value T_{max} (set to 90 seconds):

$$TQ_{phase} = min(TPQ_{phase}, T_{max})$$

After this computation, all traffic lights from the direction of the selected queue are set to green for TQ_{phase} seconds.

V. Experimental Analysis

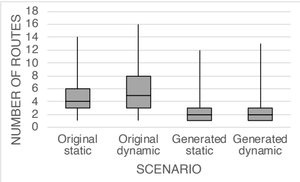
A. Experimental Setup

We ran our experiments on three different cities available in Sim^{2}Car (San Francisco, Beijing and Rome) with 200 cars each, highlighting the benefits brought by dynamic traffic lights as opposed to static traffic lights. For the 200 vehicles, we chose a small simulation area in each city, because we wanted to test with highly-congested traffic. An advantage of Sim^{2}Car in this situation is that, even if the number of vehicles is high and many computations must be performed, it behaves extremely well in terms of processing time. In order to compare the performances of the two solutions, we selected four per-vehicle metrics: routes completed, time needed to reach all the destinations, average speed, and fuel consumption. For the latter metric, we employed the model

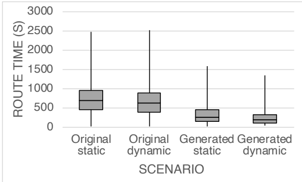
[3] All these time values were set based on experiments in the simulator, but we acknowledge that a dynamic solution would be better suited, which is something that we wish to pursue in future work.

[4] In this situation, we assume that a vehicle will not turn around in an intersection, but this is something that we wish to address in the future.

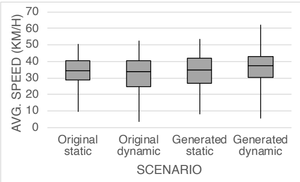
[5] This can further be extended by connecting multiple neighboring intersections and dynamically changing the waiting time.



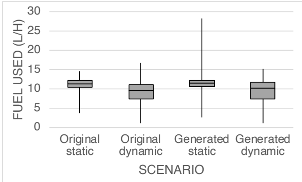
(a) Completed routes



(b) Route duration

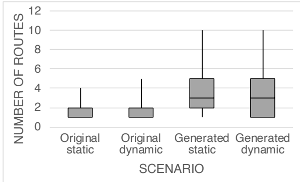


(c) Average driving speed

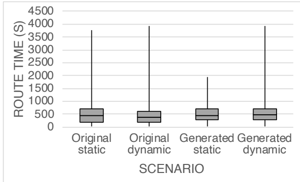


(d) Fuel consumption

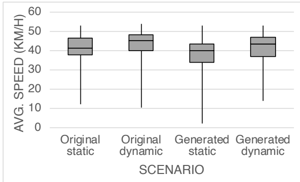
Fig. 2. Results for the San Francisco scenario.



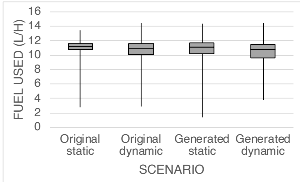
(a) Completed routes



(b) Route duration



(c) Average driving speed



(d) Fuel consumption

Fig. 3. Results for the Beijing scenario.

proposed in [16]. The metrics were chosen in order to highlight the benefits of our solution in terms of driver benefits and green transport, which were the goals of our proposed dynamic traffic light system. Because Sim^{2}Car only considers one type of vehicle, the model in this simulation was adjusted for light vehicles. Thus, the quantity of fuel Δf consumed in a time interval is computed as:

$$\Delta F = (f_i + \beta_1 R_t v + [\frac{\beta_2 M_v a^2 v}{1000}]_{a>0})\Delta t, R_t > 0$$

$$\Delta F = f_i \Delta t, R_t \leq 0$$

In the formulas above, v is the vehicle's velocity, a is its acceleration, M_v its mass, R_t is the total force acting on a vehicle, f_i is the idle fuel consumption, β_1 is an efficiency parameter and β_2 is an energy-acceleration efficiency parameter. Based on the work reported by Akcelik and Besley [17], we set the vehicle-specific parameters as follows, in order to simulate a light vehicle: M_v was set to 1100 kg, f_i to 1350 mL/h, β_1 to 900 mL/kJ and β_2 to 300 mL/(kJ $\times$ m/s^2).

We measured the four metrics on three city-wide scenarios, in San Francisco, Beijing and Rome, both on original taxi traces (from CRAWDAD[6]) and using the Sim^{2}Car Trace Tool, which allows the user to select a rectangle on a map and then generate traffic inside it. We did not settle only for original traces because the data are not entirely complete, since they were collected only when the taxi meters were running, and real traffic lights also had an impact on the behavior of the vehicles. We simulated the traffic under two different conditions: with static traffic lights (each traffic light has a fixed time set) and with dynamic traffic lights (the color of each traffic light is changed based on the number of waiting cars, as presented in Section IV).

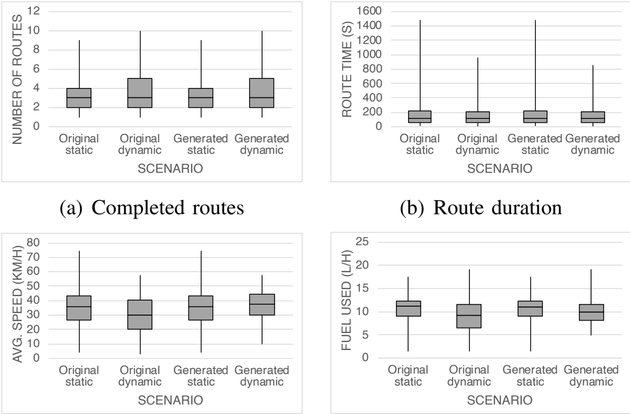
B. Results

Figure 2 shows the results obtained for the San Francisco scenario. In Figure 2(a), it can be observed that the dynamic traffic light solution is able to improve the number of completed routes for both original and generated traffic. For the former, the dynamic algorithm is able to increase the number of completed routes by 29%, which translates to more than 1 extra route per simulation. For Sim^{2}Car-generated traffic, the improvement brought on by dynamic traffic light scheduling is 17.4% in terms of number of routes finalized. It should be noted here that the metric was not improved for every individual vehicle, because Sim^{2}Car uses an intelligent routing algorithm for the car module, which may route a vehicle in an area with more traffic lights than in the static light scenario. However, it is important that the overall number of completed routes was improved by our solution in both cases, because per-vehicle metrics end up being balanced in time, thus allowing all drivers to benefit. It is also interesting to note that the number of routes is much higher for original (i.e., real-life) traces, which is caused by the fact that, in reality, taxi rides vary and can also span short distances and durations. An improvement can be observed when analyzing the average route duration per vehicle, depicted in Figure 2(b). Dynamic traffic lighting reduces the average route time by 8% on original traces and by 19% on generated traces (amounting to a reduction of about a minute for both test cases).

In terms of average speed and fuel consumption, the results shown in Figure 2(c) and Figure 2(d) are also positive. Due to a good synchronization of traffic lights and thus less stops and starts, the fuel consumption is reduced for both the original and the generated scenario by 16.3% and 9.9%, respectively. For both cases, this amounts to more than 1 liter per hour. When analyzing the average speed in Figure 2(c), it can be observed that the speed for the original scenario is reduced. However, even if this happens, the number of routes, route duration and fuel consumption are all improved (as previously shown), which means that a lower speed does not affect those metrics in this situation. This is caused by the fact that a vehicle might have to wait at more traffic lights than in the static test case, but the wait times will be shorter.

In the Beijing scenario, shown in Figure 3, it can be ob-

[6]https://crawdad.org/index.html.



(a) Completed routes (b) Route duration

(c) Average driving speed (d) Fuel consumption

Fig. 4. Results for the Rome scenario.

served that, while the number of completed routes is increased for both original and generated movements, the average route duration is also increased for the static scenario with 26 seconds. However, this is caused by the increase in completed routes, because more routes mean extra movement for those particular vehicles. Furthermore, Beijing has more traffic lights that are closer to one another than the ones in San Francisco, which means that extra routes naturally translate to higher traffic light waiting times. On the other hand, for the original traces (which are more realistic), the dynamic traffic lights solution is able to reduce the average route duration by 8%. Additionally, Figure 3(c) shows that the average speed is increased by 2% and 12.3%, whereas the fuel consumption seen in Figure 3(d) is reduced for the original traces (by 9.4%, i.e., about one liter) but increased for the generated scenario. However, the increase is relatively insignificant (2.1%) and is probably caused by the increase in speed, coupled with the high number of traffic lights in Beijing.

Finally the results for Rome (presented in Figure 4) show that the improvements seen in San Francisco and Beijing also apply here. More specifically, the dynamic routing solution is able to increase the number of routes by 21.5% and 21.3%, while also reducing the average route duration by 5.7% and 8.5%. Even if the average speed decreases by 11.5% for the original trace, the fuel consumption is reduced for both scenarios, with 3.9% and 5.1%, respectively. It should also be noted that Rome has fewer traffic lights compared with Beijing and San Francisco.

VI. Conclusions and Future Work

The aim of this paper was to improve the traffic of crowded cities through dynamic lights in intersections, based on data collected from vehicles. For this reason, we proposed and implemented a novel solution based on dynamic traffic light switching using context information, and showed that it is able to improve the number of completed routes, route duration, driving speed and fuel consumption for most cases. In the future, we wish to extend the proposed solution to not only use the crowdsensed information for dynamically setting the traffic lights, but also for routing traffic away from congested areas.

Furthermore, since in this paper we evaluated our proposal only against static behavior (i.e., without dynamic switching of traffic lights), in the future we would like to compare our solution to similar mechanisms, in order to better highlight its advantages.

References

[1] Hong Kong Transport Advisory Committee, *Report on Study of Road Traffic Congestion in Hong Kong*, 2014 (accessed April 12, 2019). [Online]. Available: https://www.thb.gov.hk/eng/boards/transport/land/Full_Eng_C_cover.pdf

[2] R. Jia, P. Jiang, L. Liu, L. Cui, and Y. Shi, "Data driven congestion trends prediction of urban transportation," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 581–591, 2018.

[3] Campaign for Better Transport, *Goings backwards: the new road programme*, 2012 (accessed April 12, 2019). [Online]. Available: https://bettertransport.org.uk/sites/default/files/research-files/Roads_to_Nowhere_October2012_web_spreads_0.pdf

[4] C.-S. Stoica, C. Dobre, and F. Pop, "Realistic mobility simulator for smart traffic systems and applications." in *ECMS*, 2014, pp. 530–537.

[5] S. Faye, C. Chaudet, and I. Demeure, "A distributed algorithm for multiple intersections adaptive traffic lights control using a wireless sensor networks," in *Proceedings of the first workshop on Urban networking*. ACM, 2012, pp. 13–18.

[6] M. Khalid, S. C. Liang, and R. Yusof, "Control of a complex traffic junction using fuzzy inference," in *2004 5th Asian Control Conference (IEEE Cat. No. 04EX904)*, vol. 3. IEEE, 2004, pp. 1544–1551.

[7] M. Collotta, L. L. Bello, and G. Pau, "A novel approach for dynamic traffic lights management based on wireless sensor networks and multiple fuzzy logic controllers," *Expert Systems with Applications*, vol. 42, no. 13, pp. 5403 – 5415, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417415001104

[8] Z. Cao, S. Jiang, J. Zhang, and H. Guo, "A unified framework for vehicle rerouting and traffic light control to reduce traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1958–1973, July 2017.

[9] R.-I. Ciobanu, C. Negru, F. Pop, C. Dobre, C. X. Mavromoustakis, and G. Mastorakis, "Drop computing: Ad-hoc dynamic collaborative computing," *Future Generation Computer Systems*, vol. 92, pp. 889–899, 2019.

[10] R.-I. Ciobanu, R.-C. Marin, C. Dobre, and V. Cristea, "Trust and reputation management for opportunistic dissemination," *Pervasive and Mobile Computing*, vol. 36, pp. 44–56, 2017.

[11] M. S. Ahmed, M. A. Hoque, and P. Pfeiffer, "Comparative study of connected vehicle simulators," in *SoutheastCon 2016*. IEEE, 2016, pp. 1–7.

[12] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of sumo-simulation of urban mobility," *International Journal On Advances in Systems and Measurements*, vol. 5, no. 3&4, 2012.

[13] J. Härri, M. Fiore, F. Filali, and C. Bonnet, "Vehicular mobility simulation with vanetmobisim," *Simulation*, vol. 87, no. 4, pp. 275–300, 2011.

[14] A. Horni, K. Nagel, and K. W. Axhausen, *The multi-agent transport simulation MATSim*. Ubiquity Press London, 2016.

[15] C. Gorgorin, V. Gradinescu, R. Diaconescu, V. Cristea, and L. Ifode, "An integrated vehicular and network simulator for vehicular ad-hoc networks," in *Proceedings of the 20th European Simulation and Modelling Conference*, vol. 59, 2006.

[16] C. Dobre, "Using intelligent traffic lights to reduce vehicle emissions," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 9, 2012.

[17] R. Akcelik and M. Besley, "Operating cost, fuel consumption, and emission models in aasidra and aamotion," in *25th conference of australian institutes of transport research (CAITR 2003)*. University of South Australia Adelaide, Australia, 2003, pp. 1–15.