Influence of Radio Communications on Multiple Intersection Control by a Wireless Sensor Network

Sébastien Faye, Claude Chaudet, Isabelle Demeure

***Abstract*—In this paper, we describe and enhance TAPIOCA (*distribuTed and AdaPtive IntersectiOns Control Algorithm*), a distributed algorithm that relies on a wireless sensor network to manage urban road traffic efficiently. We detail TAPIOCA with a special emphasis on communications and we study its reaction to losses and delays induced by the use of wireless communication. We then propose a prediction mechanism that alleviates these issues and show, using co-simulation between SUMO and OMNeT++, that such interpolation mechanisms are effectively able to replace missing or outdated data.**

I. Introduction

In a traditional urban road network, most intersections are either managed entirely locally, or by a central control center. The sequence and durations of the green lights are often pre-determined and do not adapt dynamically to the traffic conditions. Using the same policy for all traffic levels seems, however, sub-optimal. In previous contributions ([1], [2]), we have proposed TAPIOCA (*a distribuTed and AdaPtive IntersectiOns Control Algorithm*), an algorithm that computes and applies a green lights policy based on measurements from sensor nodes deployed on the road. Simulation results show that TAPIOCA is able to efficiently reduce the traffic load, as well as the users journey time compared to other strategies[1].

Previous contributions aimed at demonstrating the potential of a distributed sensors-based approach. We considered that the communication layers were idealistic. In this article, we study how TAPIOCA behaves over a more realistic communication network by coupling the SUMO road traffic simulator and the OMNeT++ communication network simulator, thanks to the Veins framework. The use of a real communication network simulator allows us to test the effect of congestion, delay and packet losses on the algorithm. We then propose and test appropriate reaction mechanisms.

After giving a few precisions on the scenario and a few definitions in section II, we review related works in section III. We then present the improved version of TAPIOCA with a special emphasis on the communication aspects in section IV. We then show the efficiency of TAPIOCA by simulation in section VI before concluding the paper.

II. Scenario and definitions

We place ourselves in the classical intersection model used in several academic works, composed of four *directions*, as represented on Figure 1. We suppose that people drive on the right side of the road. Each direction has several incoming lanes. Vehicles turning left use the leftmost lane, while the rightmost lane is for vehicles going straight or turning right. At each intersection, a *controller* defines and enforces a sequence of green and red lights called a *cycle*. A cycle is composed of successive periods called *phases*. In each phase, a subset of the lights is it green during a certain time, allowing some *movements* to occur simultaneously. Each movement is represented by the cardinal directions of its origin and destination (e.g. NE for vehicles coming from north and going east). A phase is defined by a set of allowed movements and a time length. We suppose that a traffic light controls each movement or, at least, each direction.

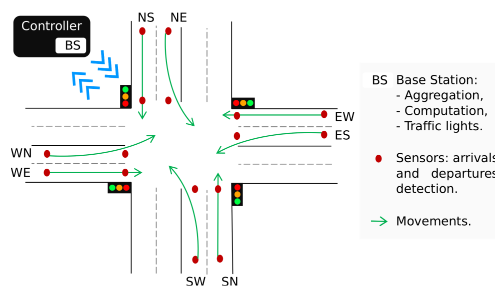


Fig. 1: A 4-lanes intersection monitored by 2 sensors per lane

The sensor nodes that monitor such an intersection are equipped with a magnetometer, which measure the changes on Earth's magnetic field to detect vehicles. They are able to detect 99 % of the vehicles that pass over them [3]. Using these devices can lead to better results than induction loops [4]: they are responsive, small, easy to install and can therefore be deployed densely, multiplying the number of measurement and action points. Knaian [5] evokes a manufacturing cost lower than $30 per unit with a 16-bit micro-controller and a size comparable to a coin. Traffic on incoming lanes is generally monitored by two sensors [6]: one located close to the traffic light, counting the vehicles *departures* and another one placed at an appropriate upstream distance, counting vehicles *arrivals*. Sensor nodes equipped with a wireless transmitter can exchange this information directly and monitor the traffic evolution at a given intersection, but are also able to communicate with close intersections and to route traffic, creating a classical multihop *wireless sensor network* (WSN) through which any intersection may acquire and send information to all relevant intersections in order to solve a given situation locally and quickly.

III. Related works

Authors using similar sensors deployment ([7], [6], [8], [9], [10]) generally feed a queueing model but use restric-

All authors are with the Institut Mines-Telecom, Telecom ParisTech, CNRS LTCI UMR 5141, Paris, France (e-mails: {first.last}@telecom-paristech.fr).

[1]Extensive simulation results are available at: `http://simt.sfaye.com`

978-1-4799-0846-2/13/$31.00

tive assumptions (e.g. all vehicles have the same speed and length [8], [9]) or do not consider the starvation scheduling problem [7], [6], [10]. Moreover, sensors are usually used as simple detectors either report measurements to a central entity, or are used locally without synchronization with close intersections. The centralized organization does not scale and has a low fault tolerance – the controller is a single point of failure –, requires an important communication volume and imposes a high decision latency. The localized approaches are non-optimal, as an intersection cannot offload traffic efficiently without knowledge of the its neighbors load.

Works issued from the distributed artificial intelligence field usually require complex calculations or deviate from the reality capabilities. Multi-agent or vehicular-based systems (e.g. [11], [12]) usually has challenging assumptions, e.g. the obligation for all the vehicles to have embedded sensors. Genetic or reinforcement learning algorithms (e.g. [13]) are frequently based on combinatorial optimization and therefore requires complex calculations and a large data history. Finally, fuzzy logic solutions (e.g. [10]) apply simple theories but empirically and without proof of stability and robustness.

IV. TAPIOCA

[1] introduces TAPIOCA, a distributed algorithm that defines the green light sequences and durations for one isolated intersection or a set of adjacent intersections. [2] improves TAPIOCA, defining in particular a new green wave creation process (path of successive green lights) and showing its efficiency through several simulations. The focus of this present article is to study the communication aspects of TAPIOCA, therefore this section presents its WSN architecture and its communication patterns. We refer the reader to [2] for a detailed description of the algorithm steps.

A. Network architecture

At a given intersection, we rely on multiple sensors with different roles: the *destination nodes* (DN) nodes measure departures when the light is green. They are located at the traffic light on each input lane. The *source nodes* (SN) measure arrivals continuously. They are located at a fixed distance of the light, not too far away, to avoid errors due to lane changes or considering new arrivals too early. There have been several contributions on the appropriate distance between SN and DN nodes and we will rely on the literature ([14], [8]) and previous contributions (e.g. [1]) that relate it to the maximum authorized green time T_{max} or that select a fixed distance, generally set to approximately 75 m.

Among the DN nodes, one *direction aggregator* is elected per direction to collect, process and aggregate the traffic of all DN nodes located on lanes that come from the same direction. These aggregators then report to a single sensor that plays the role of decision point. This decision point is a function rather than a node and could be placed on any node elected among the sensors, or even remote. It is in charge of computing the schedule according to TAPIOCA and communicating it to the controller.

This organization, represented on Figure 2, presents certain advantages. First, direction aggregators possess the information on the total number of vehicles going out of the intersection

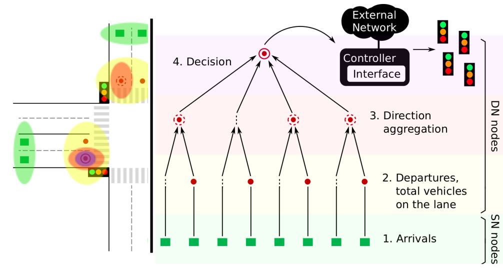


Fig. 2: A 4-lanes intersection and its corresponding architecture

in a given destination. They are able to send this information to all the next intersections and are good candidates to act as gateways towards these neighboring intersections if the network functions in a multihop manner. Second, all the higher level roles can be passed from sensor to sensor. A distributed election protocol (which can run in $\log(n)$ time [15]) can be in charge of selecting the best candidates, for example based on nodes capabilities (memory, computation power, etc.).

B. Algorithm and communication patterns

TAPIOCA works on a phase-by-phase basis. Its main goal is to select the movements composing phase $P+1$ at the end of phase P. The full algorithm is detailed below:

1) **Counting the vehicles (SN and DN nodes):** for each lane i, every DN node (layer 2) computes the number of vehicles N_i based on the departures it monitors and on the arrivals count sent by his SN node (layer 1). It then transmits this value to the direction aggregator (layer 3) node that manages the appropriate incoming direction.

2) **Per-direction aggregation (direction nodes):** for each direction a, the aggregator receives and aggregates data from the DN nodes located on a. It will transmit eventually this aggregated data to the decision node. It also forwards to the decision node the aggregated data coming from neighbor intersections. These messages are called *synchronization messages* hereafter. A synchronization message sent from an intersection I_1 to one of its adjacent intersection I_2 contains the number of vehicles going from I_1 to I_2, the total vehicles on I_1, an estimation of the time required by these vehicles to reach I_2 and the current timestamp.

3) **Next phase composition (decision node):** for each movement starting from a direction a, the master sums the N_i values received from every relevant lane to get $N^{(a,b)}$, the total queue length for the movement (a, b). If one lane i is the origin of M movements ($M > 1$), we need to avoid considering M times the vehicles of the lane i. That's why, by default, $N^{(a,b)} = \frac{N_i}{M}$ for each movement (a, b). If additional sensors are installed on the output lanes, they can cooperate with the DN node of the lane having multiple movements to determine a coefficient for each movement. For example, if an average of 60% vehicles of the lane i follow the movement (a, b), then $N^{(a,b)} = 0.6 \cdot N_i$.

Based on this data, the decision node computes a score $S(a, b)$ for each movement (a, b). This score aims at reducing *average waiting time* (AWT) while limiting starvation probability. If no vehicle is present on the lanes originating at direction a, $S(a,b) = 0\ \forall b$. Otherwise, $S(a,b)$ is a linearly weighted combination of two metrics that reflect the number of vehicles present on each lane and the time since the last selection of each movement, combining performance and fairness considerations. See [2] for further details.

Once the movements have been associated to a score, TAPIOCA examines which movements can be combined. To this extent, it uses a conflict matrix that indicates the movements that can safely be performed simultaneously. The selected phase is the combination of movements that achieves the highest score. At this stage, additional criteria can be considered (e.g. emergency vehicle detection, combination avoiding in case of accident detection).

4) **Next phase duration (decision node):** the lifetime of an intersection is, in TAPIOCA, a succession of phases. Unlike most strategies, there is no explicit notion of cycle, the phases succeed and TAPIOCA ensures that all movements are selected regularly. Once movements are selected, the controller computes the phase effective green time, based on the number of vehicles of the largest selected incoming lane and the headway time that separates two vehicles passages ([14], [2]). We limit the phase duration to T_{max}, settable by the user to balance performance and users experience aspects.

5) **Transmission to neighbor intersections (decision node, direction aggregators):** once the decision node has defined the next phase composition and timing, it transmits this information to the aggregator nodes, alongside with a possible intersection synchronization message to the less loaded neighbor intersections (in term of vehicle number).

6) **Phase application (decision node):** the decision node instructs the controller to turns the specified lights on for the specified time. This marks the beginning of phase $P+1$.

7) **During the phase $P+1$ (decision node):** if a synchronization message arrives and is considered relevant, the decision node restarts the phase selection process, forcing the selection of the movements involved to create green wave.

8) **Inter-intersection vehicles monitoring (DN nodes):** the time required to go from one intersection to the other can be estimated during phase $P+1$. The DN nodes of selected directions may send the vehicles timed signatures to the DN nodes of the next intersections that should see these vehicles pass.

V. Using Wireless Communication

TAPIOCA can run on any type of communication medium. However, if intersections controllers are sometimes linked together by optical fibers, the link with the sensors are more likely to be wireless links, as it eases deployment.

Among the suitable wireless technologies, IEEE 802.15.4 [16] is a short-range and low cost communications technology that provides coverage of 50 m to 100 m in the 2.4 GHz band for a maximum data rate of 250 Kbps and low power consumption [17]. IEEE 802.11p [18] (WAVE – Wireless Access in Vehicular Environments) also represents a viable candidate, as it is expected to be deployed in numerous vehicles and infrastructure devices. It operates in the 5.9 GHz DSRC band. 433 MHz solutions are also interesting, as they provide low-throughput and high range communication interfaces. All these technologies share some common characteristics in terms of performance and issues. They yield to similar network designs: the communication network formed has all the characteristics of a wireless local area network and may be used as a multihop network, or interconnected through WAN interfaces. We chose to study here the performances of an IEEE 802.15.4-based WSN, but the conclusions remain valid in other scenarios.

A. Delays and transmission problems

TAPIOCA relies on a regular transmission of vehicles counts, to ensure that after some time (if not immediately), the decision node receive information. In a WSN, the packet loss due to interferences or congestion can cause real issues: data may be missing or delayed such that it is outdated when it finally reaches the decision node.

Based on a co-simulation, we evaluate our architecture in Section VI and show that significant delays can be generated by the WSN and can cause several errors in accounting operations. On average on our simulation and depending on the timers values and the interferences the communications generate, the time between the generation and usage of measurements can vary from 1 s ($\Delta_{SN} = \Delta_{DN} = 1$) to 46 s ($\Delta_{SN} = \Delta_{DN} = 64$). The accounting error observed when the decision node compute a new phase can vary depending on the delays from 6.6 % ($\Delta_{SN} = \Delta_{DN} = 1$) to 56.5 % ($\Delta_{SN} = \Delta_{DN} = 64$), causing a significant loss in the user waiting time. We therefore need to modify TAPIOCA to make it resilient to losses and delays to some extent.

B. Predicting missing data

To alleviate the effect of packets losses and delays, we enhance TAPIOCA with a prediction mechanism that replaces missing data with interpolated values when necessary. In the algorithm described above, the nodes exchange periodically vehicles counts, which reflect what happens between two moments. To let nodes interpolate missing values, each SN and DN node maintains and transmits in addition an estimate of the *rate* of vehicles it counts. Let us denote by Δ_k the date at which the k^{th} vehicle is detected by such a sensor. The time $\tau_k^{(i)}$ that separates two such events at node i can be averaged using an EWMA filter (α is the filter parameter):

$$\tau_k^{(i)} = \alpha \cdot (\Delta_k - \Delta_{k-1}) + (1-\alpha) \cdot \tau_{k-1}^{(i)} ,$$

Any SN node i can then maintain an average arrival rate $\lambda_i = 1/\tau_k^{(i)}$ that it transmits with real vehicles count to its DN node and any DN node j can maintain an average departure rate, $\mu_j = 1/\tau_k^{(j)}$, that it transmits to its aggregator.

1) Prediction at DN nodes: DN nodes expect to receive data from SN nodes periodically, every Δ_{SN} seconds, which should be smaller than a phase duration. The effect of Δ_{SN} value will be studied in the simulations section. As nodes work asynchronously and as clock drifts and medium access can add delays to the message transmission, DN nodes define for each SN node a timer, whose expiration is programmed after δ_{SN} and that is reset every time a message is received from node i. When the timer expires, the DN node concludes that a data was lost and interpolates $N_i^A(t)$, the new number of arrivals value at the current date t based on the previous recorded value $(N_i^A(T_{SN}))$, on λ_i and on T_{SN_i}, the last time a message from node i was received:

$$N_i^A(t) = N_i^A(T_{SN_i}) + (t - T_{SN_i}) \cdot \lambda_i \, .$$

If δ_{SN_i} is generally larger than Δ_{SN} to deal with packet losses and excessive delays, it can also be set to a value lower than Δ_{SN}, to provide intermediate estimates on the number of vehicles, smoothing the vehicles count this way.

When δ_{SN} expires or when a message is received from a SN node, the DN node then computes $N_i = N_i^A - N_i^D$ based on N_i^A and the total departures on i, N_i^D. Finally, it send four informations to his direction aggregator (layer 3) node that manages the incoming direction DN belongs to: N_i, λ_i, μ_i and the current time t.

The direction aggregator maintains a timer $\Delta_{DN} \geq \min\{\Delta_{SN}, \delta_{SN}\}$. When this timer is reached, it transmits the last aggregated data to the decision node.

2) Prediction on the decision node: the second level prediction is on the decision node and estimates a recent value of the number of vehicles, based on the last information it has. The decision node first adjusts the number of vehicles value it has by estimating the new arrivals for each DN node i. If we denote by $N_i(t)$ the estimate of the number of vehicles in queue at the current date t and by $N_i(T_i^{DN})$ the last arrivals count received from node i at date T_i^{DN},

$$N_i(t) = N_i(T_i^{DN}) + \left(t - T_i^{DN} - T_S\right) \cdot \lambda_i \, ,$$

The startup time, T_S only has an influence for extremely high transmission frequencies. When the light for the relevant movements has been green since the last measurement (and only in this case), it estimates the departures similarly:

$$N_i(t) = \max\left(N_i(T_i^{DN}) - \left(t - T_i^{DN} - T_S\right) \cdot \mu_i\right) \, .$$

C. Fault tolerance

The prediction mechanisms described above can tolerate a few packet losses, but are inefficient when a sensor has failed, since the rates values are not updated anymore. There is no method to predict permanently missing data in the general case. Performing a correlation analysis on the different λ_i and μ_i, it should be possible to replace missing values by a linear combination of other values, but such a strategy is not guaranteed to work in every case. Similarly, data coming from neighbor intersections could also be considered. In any case, as several sensors are deployed, nodes can monitor each other and the failure of a node will be detected and reported to the controller that can then choose such a strategy, or fall back to a static plan in the worst case.

D. Managing multiple intersections

The network between intersections can rely on optical fibers, or on other radio technologies. However, when multi-hop networks are used, the inter-intersections traffic shares the wireless channel with the intra-intersection traffic. It increases, in this case, the congestion and loss probability in the communication network, especially when using a limited bandwidth radio communication technology. That's why we choose to limit the transmission of the synchronization messages (Sec. IV-B) to the only case when the receiving intersection is less loaded (i.e. has a smaller total number of vehicles) than the emitting intersection. It arrive asynchronously at an intersection that could have to deal with multiple requests during the same phase. In this case, it selects the strategy that maximizes the expected number of vehicles that will cross the intersection during the next phase by adding a constraint on the concerned movements.

Multiple intersection operations and constraints are the same as those discussed in Section V-A. However, they are amplified: if adjacent intersections are sufficiently close, there are more interference and therefore packet loss. Since TAPIOCA lets each intersection take its own decisions locally – as if it were isolated – and try to synchronize adjacent intersections with synchronization messages, packet loss does not compromise the network. It causes, in the worst case, a local decision instead of an attempt to create green wave. Moreover, synchronization messages are already affected by our prediction mechanism and are attached to a timestamp: the receiving intersection therefore cannot generate quantity or timing error.

E. Summary

Figure 3 summarizes TAPIOCA operation and Table I summarizes all communications that occur between sensors. The values of the transmission frequency and payload size are average values that reflect our simulations, however these values can be adapted at will to a particular scenario.

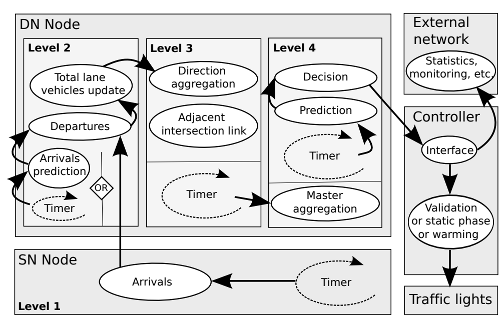
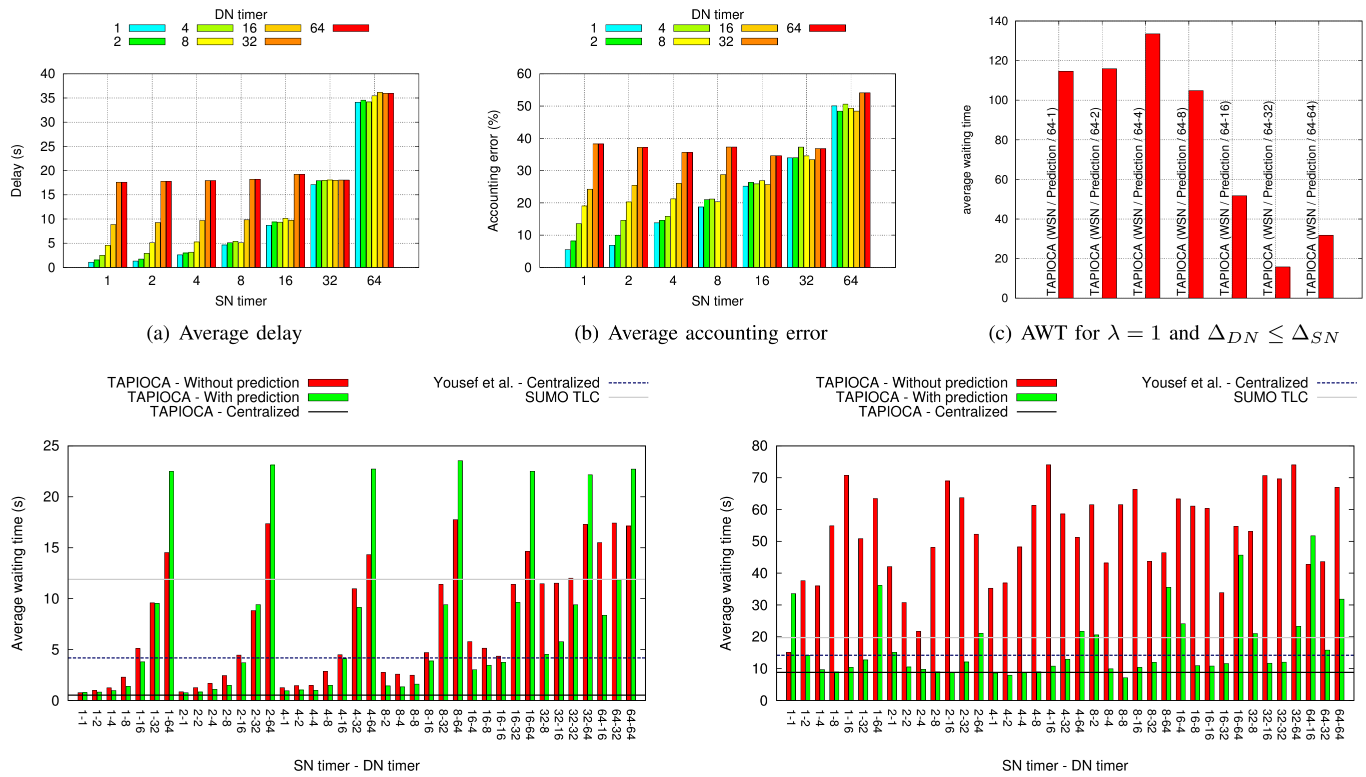


Fig. 3: Representation of our distributed algorithm

TABLE I: Communications summary

Message	Source and dest nodes	Frequency	Payload size
Arrivals [(A)]	SN $\rightarrow$ DN	Every Δ_{SN}	56 bytes
Vehicles count	DN $\rightarrow$ aggreg.	Every δ_{SN} or when [(A)] arrives	184 bytes
Aggregated vehicles count	aggreg. $\rightarrow$ decision	Every Δ_{DN}	472 bytes
Next phase characteristics	decision $\rightarrow$ {controller ; aggregators}	1 pkt/phase	64 bytes
Synchronization	aggreg. $\rightarrow$ neighbor aggreg.	1 pkt/phase (asynchronous)	200 bytes
Vehicles sign. (optional)	DN $\rightarrow$ neighbor DN	1 pkt/phase (asynchronous)	108 bytes



(a) Average delay

(b) Average accounting error

(c) AWT for $\lambda = 1$ and $\Delta_{DN} \leq \Delta_{SN}$

(d) AWT for $\lambda = \frac{1}{3}$

(e) AWT for $\lambda = 1$

Fig. 4: Simulations on one isolated intersection

VI. SIMULATIONS

We evaluate TAPIOCA by using co-simulation between SUMO 0.17[2] and OMNeT++ 4.2[3], linked together by the Veins 2.1 framework[4]. The nodes are deployed according to the architecture described on Figure 1 and use IEEE 802.15.4 *(non-beacon enabled)* standard for communication, using the CSMA protocol. Their network interface controller implements a Texas Instruments CC 2420 802.15.4 network interface card.

Figures 4 and 5 present evaluations performed on one and ten adjacent intersections respectively, randomly generated by SUMO with a low traffic intensity ($\lambda = \frac{1}{3}$ vehicles per second) and a higher traffic intensity ($3 \cdot \lambda$). Each simulation ran during 3,600 s. The complete results and other scenarios (e.g. random or growing arrivals) are available online[5]. Our results show the comparison between five methods. First, a traffic light plan generated by SUMO, acting as a predetermined control reference. Then, an adaptive algorithm from the literature [7] and TAPIOCA: both are centralized and are not subject to the network constraints, they can therefore act as adaptive control references working regardless of the equipment (e.g. induction loops). The last two methods implement TAPIOCA distributed over WSN respectively using and not using the prediction described in V-B. These last two methods are simulated using different values for Δ_{SN} and Δ_{DN} timers: 1, 2, 4, 8, 16, 32 and 64 s, i.e. 49 possible configurations.

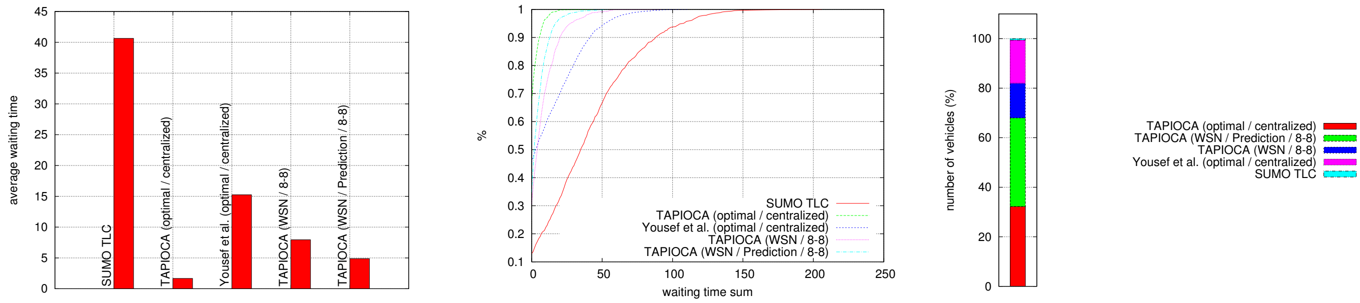
A. Performances on one intersection

Simulation results show that all implementations of TAPIOCA achieve a better AWT than the SUMO TLC (on average 71.5 % better for the centralized version) and to the Yousef et al. algorithm (59%). Figures 4(a) and 4(b) show respectively the average delay between measurements and their usage at the decision node, and vehicles count error. These figures show that these values can rise quickly depending on the parameters. According to Figure 4(c) and on average on all our simulations, we find that Δ_{DN} must be at least equal to one quarter of Δ_{SN} to ensure that the estimation does not become exaggerated and wrong. Figures 4(d) and 4(e) show the efficiency of our prediction mechanism with two different arrival flows. We see that our solution is more effective when the traffic is more important: due to the regularity of vehicles presence, estimations are therefore easier to calculate.

B. Performances on multiple intersections

In this paper, we choose to present an intermediate case scenario, by setting $\Delta_{SN} = \Delta_{DN} = 8$. Simulation results presented on Figures 5(a) and 5(b) show that all implementations of TAPIOCA still achieves better average waiting and trip time than the SUMO TLC: 93.8 % better for the centralized version, 88 % better for the WSN version with our prediction

[2]http://sumo.sourceforge.net
[3]http://www.omnetpp.org
[4]http://veins.car2x.org
[5]**http://simt.sfaye.com/WSN/**



(a) AWT (b) CDF of the AWT (c) Best strategy repartition

Fig. 5: Simulations on ten adjacent intersections with $\lambda = \frac{1}{3}$ and $\Delta_{SN} = \Delta_{DN} = 8$

mechanism and 75 % better for the basic WSN version. Figure 5(c) shows which method is the most efficient depending on the number of vehicle it achieve the best waiting time: we can see that TAPIOCA is almost always the best choice, and that the WSN-based implementation with prediction is the best solution in 40 % of the cases.

In some cases in our simulations, using TAPIOCA with a WSN and the prediction is even more efficient than the centralized version (e.g. Figures 4(e), 5(c)). This shows that our prediction method, in addition to consider transmission delays, also considers special circumstances, e.g. when the queue is too long and goes beyond the SN sensors.

VII. Conclusion and Future Works

In this paper, we propose and evaluate a distributed adaptive traffic lights control algorithm for multiple intersections that uses a WSN. We also define a four-layers architecture and show the efficiency of a WSN and the interest in managing the traffic at the intersection granularity.

In the future, we plan to complete this first communication study of TAPIOCA. For example, we plan to study energy savings and limit the WSN communications, by changing dynamically the timers based on the stability of the arrival and departure rates. Moreover, we limit here the communication between intersections to direct neighbors. However, we could benefit from information coming from a greater distance, especially when building green waves. The question of the appropriate communication distance remains open, and its answer will probably depend on the considered scenario, which would plead in favor of a dynamic approach, the communication distance being set according to the current transportation system congestion level.

References

[1] S. Faye, C. Chaudet, and I. Demeure, "A distributed algorithm for multiple intersections adaptive traffic lights control using a wireless sensor networks," in *UrbaNe Workshop (held in conjunction with ACM CoNEXT 2012)*, Nice, France, Dec. 2012, pp. 13–18.

[2] ——, "Multiple intersections adaptive traffic lights control using a wireless sensor networks," Institut Mines-Télécom, Télécom ParisTech, Tech. Rep. 2013D005, Aug. 2013.

[3] S. Cheung, S. Coleri, B. Dundar, S. Ganesh, C. Tan, and P. Varaiya, "Traffic measurement and vehicle classification with single magnetic sensor," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1917, no. -1, pp. 173–181, Dec. 2005.

[4] I. Corredor, A. García, J. Martínez, and P. López, "Wireless sensor network-based system for measuring and monitoring road traffic," in *6th Collaborative Electronic Communications and eCommerce Technology and Research (CollECTeR)*, Madrid, Spain, Jun. 2008.

[5] A. N. Knaian, "A wireless sensor network for smart roadbeds and intelligent transportation systems," Master's thesis, Massachusetts Institute of Technology, Jun. 2000.

[6] M. Tubaishat, Q. Qi, Y. Shang, and H. Shi, "Wireless sensor-based traffic light control," in *5th IEEE Conference on Consumer Communications and Networking (CCNC 2008)*, Las Vegas, USA, Feb. 2008.

[7] K. M. Yousef, J. N. Al-Karaki, and A. M. Shatnawi, "Intelligent traffic light flow control system using wireless sensors networks," *Journal of Information Science and Engineering*, vol. 26, no. 3, May 2010.

[8] B. Zhou, J. Cao, X. Zeng, and H. Wu, "Adaptive traffic light control in wireless sensor network-based intelligent transportation system," in *72nd IEEE Vehicular Technology Conference Fall (VTC 2010-Fall)*, Ottawa, Canada, Sep. 2010.

[9] B. Zhou, J. Cao, and H. Wu, "Adaptive traffic light control of multiple intersections in wsn-based its," in *73rd IEEE Vehicular Technology Conference (VTC Spring)*. IEEE, 2011, pp. 1–5.

[10] F. Zou, B. Yang, and Y. Cao, "Traffic light control for a single intersection based on wireless sensor network," in *9th International Conference on Electronic Measurement & Instruments (ICEMI 2009)*, Beijing, China, Aug. 2009.

[11] C. Wenjie, C. Lifeng, C. Zhanglong, and T. Shiliang, "WITS: A Wireless Sensor Network for Intelligent Transportation System," *First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS'06)*, pp. 635–641, Jun. 2006.

[12] D. Houli, L. Zhiheng, and Z. Yi, "Multiobjective reinforcement learning for traffic signal control using vehicular ad hoc network," *EURASIP J. Adv. Signal Process*, vol. 2010, pp. 7:1–7:7, Mar. 2010.

[13] X.-F. Chen and Z.-k. Shi, "Real-coded genetic algorithm for signal timing optimization of a single intersection," in *Machine Learning and Cybernetics, 2002. Proceedings. 2002 International Conference on*, vol. 3. IEEE, 2002, pp. 1245–1248.

[14] US Dept. of Transportation, Federal Highway Administration, Office of Operations, *Traffic control systems handbook*, 2005, http://ops.fhwa.dot.gov/publications/fhwahop06006/.

[15] T. Jurdziński, M. Kutyłowski, and J. Zatopiański, "Efficient algorithms for leader election in radio networks," in *21th annual symposium on Principles of distributed computing*. ACM, 2002, pp. 51–57.

[16] IEEE Standard for Local and metropolitan area networks, "Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)," 2011.

[17] M. Collotta, G. Pau, V. Salerno, and G. Scatá, "Wireless sensor networks to improve road monitoring," *InTechOpen*, 2012.

[18] IEEE Standard for Information technology, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications – Amendment 6: Wireless Access in Vehicular Environments," 2010.