V-DESYNC: Desynchronization for Beacon Broadcasting on Vehicular Networks

Tossaphol Settawatcharawanit, Supasate Choochaisri, Chalermek Intanagonwiwat, Kultida Rojviboonchai*
Department of Computer Engineering
Chulalongkorn University
Bangkok, Thailand
{tossaphol.s, supasate.c}@student.chula.ac.th, {chalermek.i, kultida.r}@chula.ac.th

***Abstract*—Several prospective applications on vehicular networks have been defined. Most applications rely on beaconing mechanisms to broadcast the presence and updated information of a vehicle to surrounding neighbors. However, due to the broadcasting nature, no acknowledgement mechanism is provided. Therefore, vehicles do not perceive beacon collision if two or more vehicles simultaneously broadcast the beacons. Consequently, vehicles miss updated information from their neighbors. In this paper, we propose V-DESYNC, an algorithm that distributively desynchronizes vehicles to broadcast beacons at different times based on only timing information. V-DESYNC is designed to avoid the beacon collision and tolerate the highly dynamic behavior of vehicular networks. Our evaluation results indicate that V-DESYNC can significantly reduce the number of beacon collisions without decreasing the beaconing rate on vehicular networks.**

I. Introduction

In the past few years, vehicular networks have attracted a large number of academic and industrial research communities. Several prospective safety and non-safety applications based on vehicle-to-vehicle communication have been presented such as road accident warning, pre-crash sensing, lane changing assistance, and intersection traffic management. Most of these applications rely on beaconing mechanisms to update surrounding contexts among nearby vehicles. For example, the European Telecommunications Standards Institute (ETSI) has specified the beaconing rate for safety co-operative awareness applications [1]. Normally, in a beaconing mechanism, a vehicle periodically broadcasts a beacon without reliable guarantee (*e.g.*, no acknowledgement mechanism). Therefore, if two or more beacons collide, vehicles are not able to perceive the collision. Consequently, vehicles miss updated information from their surrounding neighbors.

To mitigate the beacon collision problem, several adaptive beaconing schemes have been proposed [2]–[5]. In such schemes, vehicles adapt their beaconing rates based on surrounding contexts. However, adaptive schemes only aim to find appropriate beacon rates but not directly avoid the collision.

In this paper, we propose a novel concept for beacon collision avoidance in vehicular networks using desynchronization. Based on only listening to neighbors' firing messages, desynchronization attempts to schedule nodes to perform tasks at different time. Previous works (*e.g.*, DESYNC [6], DWARF [7]) have shown that desynchronization successfully schedules nodes in static wireless networks. However, desynchronization in vehicular networks has not been investigated. To the best of our knowledge, this paper is the first to propose a desynchronization algorithm for vehicular networks.

To apply desynchronization with vehicular networks, several challenges arise. First, due to mobility of vehicles, a desynchronization algorithm should tolerate highly dynamic networks. Second, there is a chance that two vehicles hear the same firing messages. By relying on the same received information, they adjust their firing time to the same time. Consequently, their firing messages (*i.e.*, beacons) will collide. Furthermore, vehicles in a network are heterogeneous; different vehicles may use different beacon rates to broadcast beacons depending on applications [1]. Previous works on desynchronization do not handle such heterogeneity. In this paper, we propose V-DESYNC, a desynchronization algorithm for vehicular networks. V-DESYNC uses a simple but efficient mechanism to avoid the beacon collision. Additionally, V-DESYNC is able to handle nodes with different beaconing rates and nodes with adaptive beaconing rates. Furthermore, V-DESYNC works on the application layer; therefore, no modification of the MAC layer is required.

The rest of this paper is structured as follows. In Section II, we review previous works on beaconing mechanisms for vehicular networks and on desynchronization for wireless networks. Section III describes the V-DESYNC algorithm. The Performance evaluation is reported in Section IV. Finally, Section V concludes the paper and discusses the future works.

II. Related Work

Several works adjust the beaconing rate based on surrounding contexts. CAR [4] uses a number of neighbors to adjust a beaconing rate. A high number of neighbors results in a low beaconing rate and vice versa. DECA [8], a reliable broadcast protocol, also uses a number of neighbors to adjust a beaconing rate. In [5], a node broadcasts a beacon to update its position to neighbors. Their algorithm reduces the number of beacon transmissions by using position predictions. A node includes additional data such as movement direction and velocity into a beacon. A receiving neighbor predicts the approximate position of a sender. Therefore, the sender can reduce the beaconing rate and sends a new beacon only when

*Corresponding author

978-1-4673-0990-5/12/$31.00 ©2012 IEEE

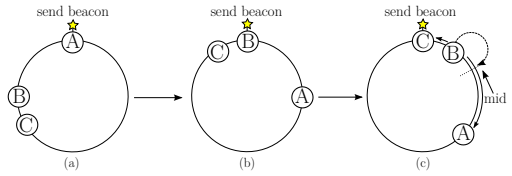

Fig. 1: DESYNC algorithm. A node adjusts its firing time to the midpoint between nodes firing just before and just after its firing.

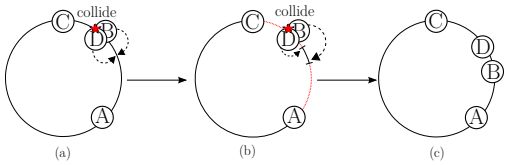

Fig. 2: V-DESYNC: firing time adjustment with a random offset. Instead of jumping to the calculated midpoint, each node jumps to a point near the calculated midpoint.

it moves away from the predicted position for a distance of a threshold value. Schmidt et al. [2] proposes an algorithm to adjust the beaconing rate based on traffic situations. In contrast, our work does not reduce the beaconing rate but directly avoids collisions based on timing information using desynchronization.

Recently, desynchronization techniques have been proposed. Desynchronization attempts to schedule different nodes do tasks at different times based on the timing information of firing messages. In DESYNC [6], for each period, a node adjusts its firing time towards a midpoint between two nodes that fire just before and just after itself to avoid a collision. DWARF [7] uses an algorithm inspired by robotic circular formation to create an artificial force field. The algorithm uses all neighbors' timing information to calculate the overall force acting on one node. Then, the node adjusts its firing time to the point of time that reduces the excessive force acting on itself. Their approach significantly reduces desynchronization error (*i.e.*, nodes are equivalently separated in the time domain) compared to DESYNC. However, relying on all neighbors' information is not appropriate in vehicular networks due to the highly dynamic characteristic. M-DESYNC [9] uses a distributed graph coloring algorithm to reserve time slots. However, the algorithm requires re-calculation when a topology changes. Therefore, M-DESYNC also does not tolerate the highly dynamic vehicular networks.

Our objective is to avoid beacon collisions in highly dynamic vehicular networks. The equivalent time interval is not a crucial requirement. Therefore, our approach is based on DESYNC because their algorithm relies on only two neighbors' timing information to avoid the collision which is more suitable to highly dynamic networks.

III. Algorithms

In this section, we describe V-DESYNC, our desynchronization algorithm for beacon collision avoidance in vehicular networks. We begin with an overview of DESYNC that our algorithm is based on. Then, we present our main algorithm.

A. DESYNC Overview

In DESYNC [6], the desynchronization framework has been introduced. The message firing period T is abstracted as a ring (see Figure 1). As the time progresses, each node moves clockwise around the ring. When a node reaches the top of the ring, a task is executed (*i.e.*, a message is fired).

To avoid collision, each node listens to their neighbors' firing messages. After a node hears firing messages just before and just after its firing, the node calculates a midpoint between two neighbors' firings. Then, the node moves towards the midpoint. In Figure 1(a), node A fires a message just before node B. Then, in Figure 1(b), node B fires a message. After that, in Figure 1(c), node C fires and node B calculates the midpoint between node A and C to move towards that point. Formally, a node calculates the midpoint ϕ_{mid} as follows:

$$\phi_{mid} = \frac{1}{2}\left[\phi_{i+1} + \phi_{i-1}\right], \tag{1}$$

where ϕ_{i-1} and ϕ_{i+1} denote the firing points of nodes that fire just before and just after node i respectively. Then, node i moves to a new point ϕ_i' based on the moving average function:

$$\phi_i' = (1-\alpha)\phi_i + \alpha\phi_{mid}, \tag{2}$$

where α is a step size that indicates how far a node moves from the previous time point to the new target time point.

B. V-DESYNC

V-DESYNC is based on the ring framework as in DESYNC. As time passes, each node moves clockwise on the time ring and fires when it reaches the top of the ring. However, V-DESYNC solves two issues of DESYNC. First, if two nodes are unfortunately on the same time phase on the time ring, they perceive the same view of the time ring. Consequently, they fire at the same time, their firing messages collide, they move to the same target point, and they fire at the same time again. Second, in vehicular networks, different nodes may use different time periods. Therefore, time rings of different nodes may be different in size. V-DESYNC handles these problems with two mechanisms: firing adjustment and multiplicative increase adaptive beaconing.

1) Firing Adjustment: The problem that two nodes are at the same phase, fire beacons at the same time, and move to the same new phase arises because there is no acknowledgement for beacon broadcasting. When two beacons collide, both senders do not notice the collision. This problem is illustrated in Figure 2(a). Node B and D are at the same point and perceive the same view of the ring. After they hear firings of node A and C, they move to the same target point.

To avoid such a problem, we propose a simple but efficient firing adjustment mechanism. As in DESYNC, each node listens to their neighbors and calculates a new firing time based on the midpoint between two neighbors' firings. However, instead of moving directly to the target point, each node moves to a point near the target point. In other words, each node

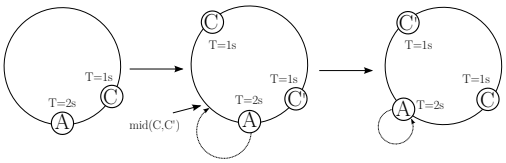

Fig. 3: Node A's view (ring size = 2s) when there is a shorter period node. Node A hears all firing messages from node C (C and C') and moves to the middle of two firing events.

randomly adds a small offset which is uniformly chosen from a continuous range to the calculated midpoint. Figure 2(b) and 2(c) illustrate this firing adjustment mechanism. With the random offset, the opportunity that two nodes move to the same new point to cause a consecutive collision can be reduced. In our implementation, we set the offset to be 10% of a time interval between two neighbors' firings.

However, to move to a point around the target is a best effort mechanism. It is not guaranteed that the next firings of two nodes will not collide again. Nevertheless, the probability that two firings consecutively collide for n rounds in a row P_n is an exponentially decreasing function as follows: $P_n = P_0^n$, where $P_0 \in [0, 1]$ is the probability that two firings collide. Therefore, it is unlikely that two colliding nodes at the current round will fire and cause the collision again at the next round. Even if they collide again, eventually, two colliding nodes will be separated away.

2) Handling Different Periods with Multiplicative Increase Beaconing: In vehicular networks, vehicular nodes are heterogeneous and usually independent to others. Therefore, different nodes can use different time periods to broadcast beacons. Furthermore, nodes are able to use an adaptive beaconing mechanism to adapt their time periods.

If there is a node with a shorter time period than others, the desynchronization algorithm is not affected because a longer period node can hear all firings from a shorter period node as illustrated in Figure 3. However, if there is a node with a longer period than others, the different time periods cause DESYNC to fluctuate. Figure 4(a) illustrates a scenario that node A and B use 2-second period whereas node C uses 4-second period. Instead of appearing multiple times on the ring as in the previous scenario, node C alternately appears and disappears in the node A's view. To simplify the explanation, we assume node B and C do not adjust their firing time.

To solve the different-periods issue, V-DESYNC includes the time value of a chosen period into a beacon. When a node receives a beacon, the node processes with the following algorithm. If the period value in the received beacon is less than its period, the receiving node knows that the firing node will appear on multiple positions of the ring and treats those positions as different nodes. In this case, the receiving node can hear all multiple firings within its own time period (as illustrated in Figure 3). If the period value in the received beacon is greater than its period, this case is more complicated because the receiving node may not hear the firing node in some round. In this case, the receiving node should reserve the time position for that firing node and treat as there is a virtual

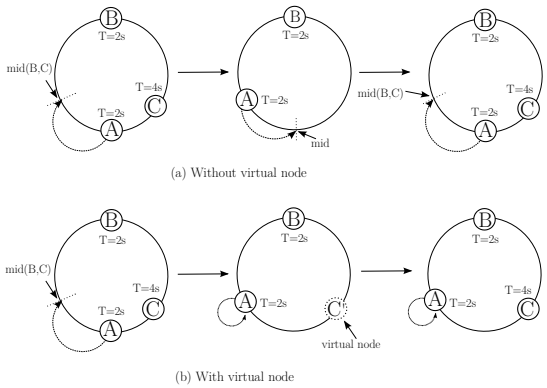

Fig. 4: Node A's view (ring size = 2s) when there is a longer period node.

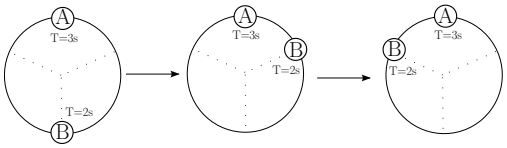

Fig. 5: Node A's view (ring size = 3s) when time periods are not multiple of each other.

node on that time position. The reason is that even the firing node disappears from the ring in the next round, it eventually will appear again in some succeeding round. Therefore, in the round that the node with longer period does not fire, the receiving node assumes the longer-period node virtually exists on the ring as demonstrated in Figure 4(b).

However, in the above examples, we assume that different time periods are multiple of each other (*i.e.*, 1, 2 and 4 seconds). If different time periods are multiple of each other, it is easy to predict the positions of each node on the ring. In a node's view, its neighbor with a shorter period appears on multiple points symmetrically whereas its neighbor with a longer period alternately appears and disappears at only one point. In contrast, if different time periods are not multiple of each other, the points on the ring for each round is complicated to predict and fluctuations occur. Figure 5 demonstrates this problem in node A's view. Again, to simplify the explanation, we assume node A and B do not adjust their firing time. Node B uses a shorter period than Node A and their periods are not multiple to each other. Consequently, node A sees node B appearing at different points in each round. Therefore, we propose the multiplicative increase beaconing to solve this problem.

In multiplicative increase beaconing, nodes set their time periods to be multiple of each other. Let T_0 be a minimum period value used in a system, k be a step-size multiplier, and m be a number of different period values allowed to be used in the system. Each node chooses any time period in the set T, where $T = \{x | x = k^i T_0, 0 \leq i < m\}$. For example, if a system sets T_0 = 1 second, k = 2, and m = 5, the allowable periods are 1, 2, 4, 8, and 16 seconds. We note that the sequence of allowable periods is a multiplicative increasing sequence. We assume that every node follows the

Parameter	Value
Simulation time	100 seconds
Number of runs	10
Number of nodes in a topology	48, 240, 720, 1440, 1920
Average number of nodes in communication range	3, 15, 41, 79, 110
Path attenuation model	2-ray Ground
Beacon size	30 bytes
Transmission range	250 m
Vehicle velocity	0-120 km/hr

TABLE I: Simulation parameter settings

same algorithm. Therefore, multiplicative increase beaconing ensures that all nodes' periods are multiple of each other.

Multiplicative increase beaconing in V-DESYNC also supports adaptive beaconing. In several adaptive beaconing approaches, a node may change its time period (*i.e.*, its beaconing rate) based on a particular event. For example, a node calculates a new time period when a number of neighbors is changed. However, the calculated period may be not in the set of allowable periods. Therefore, in V-DESYNC, a node sets its period to be an allowable period value closest to the calculated period. We can search for such an allowable value in $\mathcal{O}(\textbf{log}\ m)$ time with the binary search where m is a number of different allowable values in the multiplicative increase set.

Additionally, to deal with the dynamic behavior of vehicular networks (*i.e.*, mobile nodes can be connected or disconnected anytime), each node keeps track of other nodes by using time-to-live timers. When a node hears a firing message, the node records the firing time as a virtual node's position on the time ring and starts a time-to-live timer. The time-to-live timer is set equal to the period value in the received message. If the timer fires, the virtual node is removed from the time ring. This mechanism prevents stale virtual nodes on the ring.

In the next section, we evaluate and compare V-DESYNC to other approaches to realize performance of the algorithm.

IV. Simulation Results and Evaluations

We have conducted the performance evaluation on ns-2.34 [10]. We implemented V-DESYNC on the application layer to control the time and rate to fire a beacon. The MAC layer is non-modified CSMA/CA. We used SUMO [11] to generate mobility traces. In our evaluation, we used Manhattan 3x3 km^2 grid urban area with two-lane because high vehicle density in the urban area leads to network contention and beacon collision. Table I shows the rest of parameter settings.

To demonstrate how V-DESYNC helps reduce beacon collisions, we evaluate V-DESYNC in three schemes: homogeneous constant rate beaconing, heterogeneous constant rate beaconing, and adaptive rate beaconing. In homogeneous constant rate beaconing, all vehicles use the same beaconing rate. In contrast, in heterogeneous constant rate beaconing, each vehicle randomly chooses a constant beaconing rate, whereas, in adaptive rate beaconing, each vehicle adapts its beaconing rate dynamically.

A. Homogeneous constant rate beaconing

We compare V-DESYNC against constant rate beaconing which is used in most applications defined in [1] and

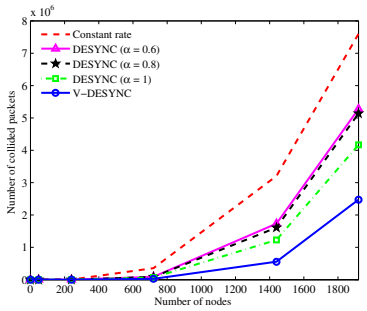

Fig. 6: Beacon collision of homogeneous constant rate T = 1 second.

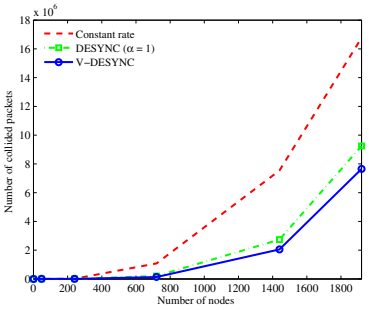

Fig. 7: Beacon collision of homogeneous constant rate T = 0.5 seconds.

DESYNC. We assume all nodes use the same beaconing rate (*i.e.*, same time period). We also vary the step size (α) to find at which step size DESYNC performs best. Figure 6 shows a number of beacon collisions for 1-second time period. As shown in the figure, the best step size for DESYNC is 1. We also use 1 as the step size for V-DESYNC because V-DESYNC is based on the adjustment mechanism of DESYNC. The result shown in Figure 6 indicates that V-DESYNC can reduce the number of collisions by 67% and 41% compared to constant rate beaconing and DESYNC respectively. The reason is that, in the constant rate beaconing, if two or more nodes unfortunately begin broadcasting beacons at the same time, their beacons will always collide. Due to no acknowledgement mechanism in broadcasting, nodes neither adapt their rate nor change their broadcasting time. In contrast, in DESYNC and V-DESYNC, nodes desynchronize their firing times to avoid beacon collision. However, in DESYNC, if two nodes unfortunately on the same time position and receive the same firing messages, they will move to the same time position and their firing messages will collide again. On the other hand, nodes, in V-DESYNC, move with random offsets to decrease the collision probability.

Figure 7 demonstrates a scenario when the beacon collision probability is increased by reducing the time period to 0.5 seconds. In this experiment and other following experiments, we use 1 as the step size for both DESYNC and V-DESYNC. As shown in Figure 7, V-DESYNC can reduce the number of collisions by 54% compared to constant rate beaconing. However, the number of collisions is reduced only by 17%

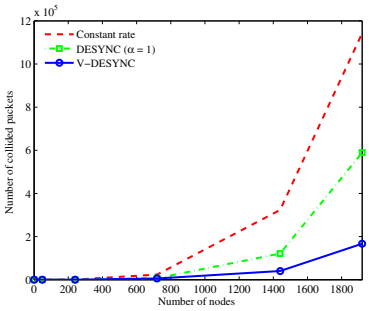

Fig. 8: Beacon collision of heterogeneous constant rate

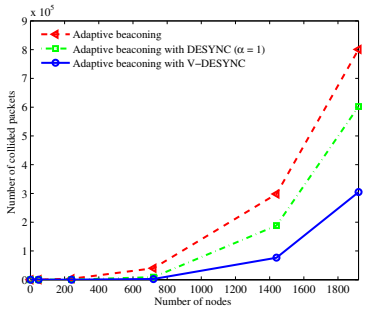

Fig. 9: Beacon collision of adaptive beaconing.

compared to DESYNC because the period is shorter. Therefore, there is not much free time interval to avoid the collision.

B. Heterogeneous constant rate beaconing

In this scenario, different time periods in a range of 1 to 4 seconds are uniformly assigned to different nodes. As shown in Figure 8, V-DESYNC reduces the number of collisions by 85% and 70% compared to constant rate and DESYNC. In DESYNC, the different time periods cause the fluctuation in firing adaptation as described in Section III-B2. A longer period node disappears from the time ring of a shorter period node. However, in V-DESYNC, the shorter period node assumes the presence of the longer period node by creating a virtual node on the ring. This technique prevents the fluctuation. As a result, the number of collisions is reduced.

C. Adaptive rate beaconing

For adaptive rate beaconing, we use a beaconing rate adaptation function that the beaconing rate is proportional to a number of neighbors which is used in several protocols such as CAR [4] and DECA [8]. The function is defined as $T = W \times n$, where T is a time period to broadcast a beacon (*i.e.*, the beaconning rate), n is a number of one-hop neighbors, and W is a constant weight. In our simulation, we set W to 0.2. We compare V-DESYNC to DESYNC and adaptive rate beaconing without desynchronization. V-DESYNC uses multiplicative increase mechanism to change the beaconing rate calculated from the same rate adaptation function. For multiplicative increase, we set T_0 (the minimum period value) to 1, k (multiplier) to 2, and m (a number of different allowable periods) to 4. Thus, in V-DESYNC, allowable periods are 1, 2, 4, and 8. The result is demonstrated in Figure 9. V-DESYNC helps reduce the number of collided beacons by 62% and 50% compared to adaptive rate and DESYNC. Even when nodes with adaptive rate beaconing dynamically adjust their rates, there is still a chance that nodes broadcast at the same time. On the other hand, nodes in V-DESYNC predict the time that other nodes will broadcast again and avoid collision.

V. Conclusion

Beaconing is a crucial mechanism for several vehicular network applications to broadcast the presence and updated information of a node. However, the beacon collision problem results in loss of important information to be used in several protocols and applications. In this paper, we propose a novel desynchronization approach for vehicular networks to avoid the beacon collision problem. We present V-DESYNC, a distributed algorithm to desynchronize nodes not to broadcast at the same time. V-DESYNC relies on only timing information and tolerates highly dynamic vehicular networks and the modification of a MAC layer is not required. Our evaluation results indicate that V-DESYNC significantly reduces beacon collisions compared to existing approaches. We believe that V-DESYNC could help increase throughput and reduce latency for several vehicular network applications because V-DESYNC not only avoids collision but also does not decrease beaconing rate. This work is the first step that applies desynchronization into vehicular networks. We will further explore potential of desynchronization on vehicular networks in our future work.

References

[1] ETSI TR 102 638, *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions.* European Telecommunications Standards Institute, 2009.

[2] R. Schmidt, T. Leinmuller, E. Schoch, F. Kargl, and G. Schafer, "Exploration of adaptive beaconing for efficient intervehicle safety communication," in *Network, IEEE , vol.24, no.1*, 2010, pp.14–19.

[3] C. Sommer, O. K. Tonguz, and F. Dressler, "Adaptive beaconing for delay-sensitive and congestion-aware traffic information systems," in *Proc. IEEE Vehicular Networking Conf. (VNC)*, 2010, pp. 1–8.

[4] V. Naumov and T. R. Gross, "Connectivity-aware routing (car) in vehicular ad-hoc networks," in *Proc. INFOCOM 2007. 26th IEEE Int. Conf. Computer Communications. IEEE*, 2007, pp. 1919–1927.

[5] A. Boukerche, C. Rezende, and R. W. Pazzi, "Improving neighbor localization in vehicular ad hoc networks to avoid overhead from periodic messages," in *Proc. IEEE Global Telecommunications Conf. GLOBECOM 2009*, 2009, pp. 1–6.

[6] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "Desync: Self-organizing desynchronization and tdma on wireless sensor networks," in *Proc. 6th Int. Symp. Information Processing in Sensor Networks (IPSN)*, 2007.

[7] S. Choochaisri, K. Apicharttrisorn, K. Korprasertthaworn, and C. Intanagonwiwat, "Desynchronization with an artificial force field for wireless networks". [Online]. Available: http://www.cp.eng.chula.ac.th/∼intanago/pub/dwarf.pdf

[8] N. N. Nakorn and K. Rojviboonchai, "Deca: Density-aware reliable broadcasting in vehicular ad hoc networks," in *Proc. Int Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON) Conf*, 2010, pp. 598–602.

[9] H. Kang and J. L. Wong, "A localized multi-hop desynchronization algorithm for wireless sensor networks," in *IEEE INFOCOM*, 2009.

[10] The Network Simulator (ns-2), http://www.isi.edu/nsnam/ns/.

[11] Simulation of urban mobility (SUMO), http://sumo.sourceforge.net/.