TIME-ANTS: An Innovative Temporal and Spatial Ant-based Vehicular Routing Mechanism*

Ronan Doolan and Gabriel-Miro Muntean, *Member, IEEE*

[1]Abstract— **Increasing amounts of time is wasted due to traffic congestion in both developed and developing countries. This has severe negative effects, including drivers stress due to increased time pressure, reduced usage efficiency of trucks and other commercial vehicles, and increased gas emissions-- responsible for climate change and air pollution affecting population health in densely populated areas. As existing centralized approaches were neither efficient, nor scalable, there is a need for alternative approaches. Social insects provide many solutions for dealing with decentralized problems. For instance ants choose their routes based on pheromones left by previous ants. However, Ant Colony Optimization is not directly applicable to vehicle routing, as routing the vehicles to the same road would cause traffic congestion. Yet, the traffic is broadly similar from work- to work-day.** *This paper introduces an ant-colony optimization-based algorithm called Time-Ants.* **Time-Ants considers that an amount of "pheromone" or a traffic rating is assigned to each road at any given time in the day. Using an innovative algorithm the vehicle's routes are chosen based on these traffic ratings, aggregated in time. After several iterations this results in a global optimum for the traffic system. Bottlenecks are identified and avoided by machine learning. Time-Ants outperforms another leading algorithm by up to 19% in terms of percentage of vehicles to reach the destination within a given time-frame.**

Index Terms—**Traffic congestion, Machine learning, Vehicle routing, VANET**

I. INTRODUCTION

Traffic congestion is a very serious problem both in developed countries and increasingly in developing countries. For instance Sao Paulo, Brazil is known to experience the world's worst traffic jams, with people stuck in traffic for an average of two to three hours at a time according to Jain et al. [1]. This report identified poor traffic management as the leading cause of traffic congestion. All countries have seen a surge in vehicle ownership; developing countries also have poor existing infrastructure and very little money to improve it. Austin et al. [2] states that developing countries are much more dependent on automobiles for transport due versatility, flexibility, and low initial cost compared with other modes of transport. At the same time pollutant levels in large developing cities' air far exceed those in the developed world. However both in developed and developing countries, the number of cars on the road is increasing without a corresponding increase in road capacity in urban areas. Despite the impact of the global recession, the number of cars produced each year has risen 33% since 1999 [3], further contributing to traffic congestion and pollution.

One potential technology to be used for alleviating traffic congestion is Vehicle Ad Hoc Networks (VANET). VANETs use individual vehicles as nodes, which communicate with each other wirelessly using Wi-Fi or cellular. This paper considers the Wi-Fi standard IEEE 802.11p, a standard specifically designed to cope with the highly mobile VANET nodes. VANETs communicate with the Internet via Road Side Units (RSU). Messages are sent from one vehicle to another or from one vehicle to a RSU using multi-hop.

Existing traffic information systems such as Tomtom [4] or Google maps traffic [5], take information from induction loops. However these are expensive and not placed at every junction. Collecting information from vehicles via VANETs could provide traffic maps with far higher granularity. VANET-based solutions could improve the efficiency of use of the road network. Traffic congestion could easily be detected and predicted without having to install many induction loops or traffic cameras. The vehicles could be instructed to avoid busy roads at certain times without having to set up large expensive electronic signs. This would benefit all the cities, but especially those which are experiencing great increase in population and number of vehicles on the road and have limited finances to invest in urban planning and/or infrastructure upgrades.

Often nature presents the best solutions to problems, by virtue of millions of years of evolution. Among these solutions, swarm algorithms address the problem of decentralized routing of units and Ant colony optimization (ACO) presents a brilliant method for choosing the best route. Each ant leaves some pheromone down when it walks along. Other ants then follow and leave down more pheromone, reinforcing the route. Although, ACO is not directly applicable to vehicle routing, as if too many cars go on the same road traffic congestion occurs, an innovative ACO extension is beneficial.

This paper proposes *Time-Ants, a novel ACO-based algorithm which determines optimum routes for vehicular traffic in both space and time dimensions.* Rakha et al. [6] showed that US vehicle traffic is very similar for weekdays, but varies considerably on weekends as long as there are no incidents. The paper by Immers et al. [7] described how the traffic flow affects congestion, namely that when increasing

the number of vehicles in a road system, a tipping point is reached at some stage. When the number of vehicles goes above this point, the average speed of vehicles rapidly drops Time-Ants takes into account both these studies and optimizes the road use.

The paper is structured as follows. Section 2 presents the related works. Section 3 describes the architecture of the system, as well as how the algorithm works and how load balancing is achieved. In section 4 the simulator model and the simulation scenario are described. In section 5 the results are presented and then commented on. Finally section 6 details the conclusions and future work.

II. RELATED WORKS

The related works are organized in three areas: VANET-based routing algorithms, inter-vehicle coordination and swarm algorithms.

A. VANET-based Routing Algorithms

Wu et al. [8] described a VANET-based routing mechanism with several different flavors. The author named this system Dynamic Navigation Algorithm (DNA). Four metrics are considered to determine how good a road is average speed on that road, inter-vehicle spacing, the road type (motorway, side street etc...) and the length of the road. The DNA flavors are described in more details in the simulation and testing scenario section.

Senge et al. [9] proposed a VANET-based algorithm based on bee swarm algorithms. The map is divided into cells. Each vehicle acts as a bee foraging along to its destination and sends a message back as to how it performed along the way. Vehicles then follow good routes from cell to cell till they reach their destination. The cellular setup makes sure information is up-to-date as it is quite local. The paper did not give enough information on the algorithm to allow its implementation and usage during testing.

Collins et al. [10] proposed a vehicle routing solution *TraffCon* which is based on the evaluation of a fitness function for each road segment. The fitness function is made up of weighted cost components consisting of road travel time, road used capacity and road fuel consumption.

Doolan et al. [11] proposed a VANET-based routing algorithm which considered traffic conditions and road conditions, such as road roughness and gradient, in order to route vehicles on a more fuel efficient route.

The advantage of Time-Ants over these solutions is the innovative consideration of historical information, which allows better prediction of future traffic conditions.

B. Inter-vehicle Coordination

Inter-vehicle coordination protocols help find the position and velocity the vehicles should have relative to each other. They do not choose which road segments the vehicles should travel on, as in the case of vehicular routing protocols. However, this area was studied to find ways of using VANETs to reduce traffic congestion.

Chuah et al. [12] described a variable speed limit scheme in order to smooth traffic flow. For instance, if there is an accident and one lane is blocked on a highway, lowering the speed limit improves the throughput, and provides a smooth flow pattern.

Ferreira et al. [13] looked at using virtual traffic lights based on vehicle to vehicle communication to improve congestion. This requires not only a high level of accuracy but also a penetration rate of 100% to prevent crashes.

Olaverri-Monreal et al. [14] gave instructions to the driver to help him/her overtake. Video streaming is used to give the driver a view of what is in front of the vehicle he/she is overtaking. Although a very interesting idea, it is difficult to simulate its effects. It would be difficult to determine with current traffic micro simulators, how driver would react to this new information source.

Reichardt et al. [15] focused on highway merging traffic. Vehicular communication would be very useful as misunderstandings between drivers cause many crashes in this scenario. For instance, typical information which could be exchanged includes vehicle trajectory, speed, length of vehicle, etc.. This idea is difficult to simulate using current traffic micro simulators.

Some ideas behind inter-vehicle coordination solutions could be adapted to rerouting algorithms like Time-Ants in order to improve travel times and safety.

C. Swarm Algorithms

A swarm is a decentralized system with quasi-homogeneous units [16]. Each unit is called an agent. These simple agents can solve complex tasks by interacting with each other. Swarm algorithms model this interactive behavior [17]. Several different types of swarm algorithms were looked at including ant-based, bee-based, firefly-based and particle-based.

1. Ant-based

Ant colony swarm theory is described by Teodorović et al. [17]. In its journey, each ant leaves a pheromone trail behind. When traveling, the ants naturally chose paths with high pheromone levels. This behavior can be abstracted as a learning algorithm, which when applied results in an optimal path.

2. Bee-based

Bees communicate via dancing (body movements). When a bee returns from a food source it communicates the distance, direction and quality to the other bees via dancing. Bees use this information to choose the optimum food source to visit. Bee colony swarm is described by Teodorović et al. [17]. Initially the bees use a random walk model to determine a partial solution. A local optimal is determined which is communicated to the other bees; this partial solution is then built on to determine a full solution.

3. Firefly-based

Fireflies light up to communicate to other fireflies. Real fireflies use this lighting communication for mating purposes, but in the firefly algorithm described in Yang et al. [18] this is not the case. All the fireflies are drawn to each other in this algorithm regardless of sex. The dimmer firefly will be drawn to the brighter firefly. The brightness

decreases over distance and if a particular firefly cannot see any others, it moves randomly. When using this algorithm to solve problems, the brightness can be proportional to a certain parameter or fitness function in order to attract the nodes to a better location.

4. *Particle swarm*

Particle swarm is based on the behaviour of bird flocks and fish schools. This set of algorithms is described by Teodorović et al. [17]. A particle remembers its best solution and that of the best solution obtained by any other particle. In the next time step, the particle moves in the direction of its best solution and the global best solution.
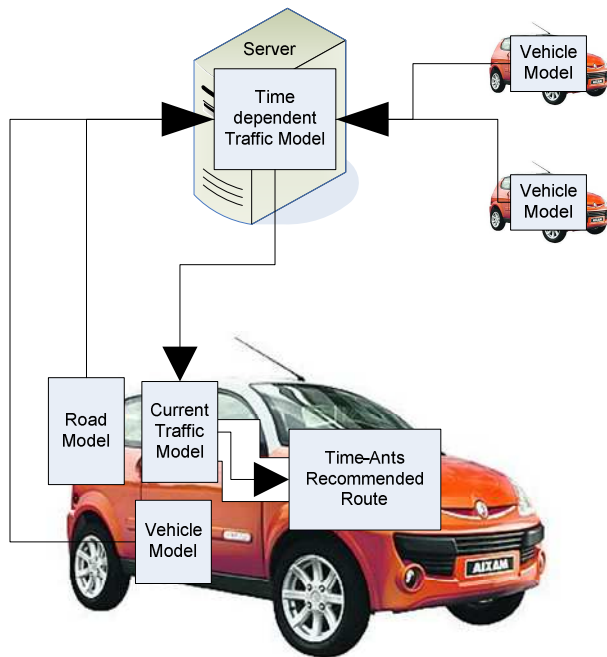

Figure 1 Time-Ants Architecture

III. TIME-ANTS ARCHITECTURE AND ALGORITHM

A. *Architecture*

Figure 1 illustrates the Time-Ants architecture, which is composed of: vehicle models, road models, a time-dependent traffic model and a current traffic model. The vehicle model contains the vehicle's speed and position. The road model contains the traffic information for a road segment that the vehicle is on. The current traffic model contains the instantaneous traffic information in the area, whereas the time-dependent traffic model contains the traffic information collected throughout a whole day.

Each vehicle is equipped with a digital map and GPS receiver to determine which road it is on. Information from these sensors is fed into the *Road Model*. The road model models the local area and the time to each junction

A server has a database with all the roads and their traffic scores, for all the times throughout the previous day. This information is used to form the *time-dependent Traffic Model*.

The vehicle communicates with the server to retrieve the traffic information on the roads at specific times, relevant to the vehicle. The vehicle sends messages containing the road and the time the vehicle will be on that road. The server then responds with the traffic rating for that road at the indicated time. This information on the traffic ratings at the required times will allow the vehicle to construct a model of the current traffic conditions, called the *current Traffic Model*. The estimated traffic on each road is then entered into the Time-Ants algorithm to compute the optimum route for the vehicle. This route is then sent to the vehicle model in order to reroute the vehicle.

A *Vehicle Model* contains information on vehicle position and speed. The current position, speed and time from all the vehicle models are sent to the server regularly. This allows the server to build a new time-dependent traffic model.

The vehicle-server communications are achieved using 802.11p and the use of RSUs. If the vehicle is not in the immediate range of an RSU, multi-hop communication, via other vehicles, is used to reach the nearest RSU.
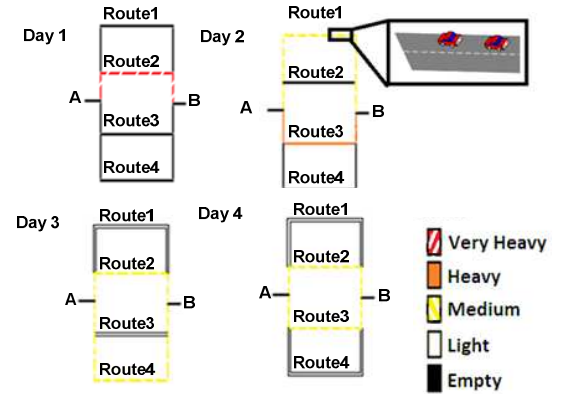
B. *Algorithm Principle*

Figure 2 Illustration of iterative improvement in Time-Ants

As traffic conditions are roughly similar from day to day during weekdays, historical data can be used to make decisions for routing and eventually the road network can be more efficiently used.

Iterative use of the Time-Ants algorithm over several days enables important benefits, as illustrated for instance in figure 2. During Day 1 Route 2 is overused, so the next day more vehicles are sent on other routes using Time-Ants. This process continues from day to day. By Day 4 the traffic is a lot smoother. Periodically, each vehicle determines the traffic rating on the road it is travelling. The road rating is determined by comparing the speed of the vehicle divided by the speed limit of the road. The vehicles then send this information over the VANET to a server. This is recorded on the central server for that location and time of day.

Each road segment then has a score or "pheromone" amount for a given time in the day. As suggested in the traffic flow theory by Immer et al. [7], vehicle speeds are only affected by traffic congestion when vehicle numbers increase above a certain point. Consequently, Time-Ants

load balancing is performed when the speed of vehicles on a road dropped beneath congestion threshold of the speed limit only.

If load balancing is not needed, the vehicles simply drive on the fastest route. The fastest route is determined by dividing the route length by the speed limit.

When choosing a route, each vehicle estimates how long it will take to get to each junction. It then sends a request to the server for any ratings associated with the adjoining roads near the time it will arrive there. After the vehicle receives the information, one of these roads is chosen according to the Time-Ants load balancing mechanism and according to the road's score. If there are no recorded ratings stored on the server around the requested time, then it is assumed that the road is empty at that time and then rating is defined as the speed limit of that road.

C. Load Balancing

Time-Ants load balancing is performed based on a metric determined from the ratio between the speed limit and the actual speed on that road as discussed in the speed/flow diagram by Immer et al. . Time-Ants applies load balancing when the actual speed drops beneath 80% of the speed limit.

The load balancing algorithm should prevent flash crowding on the roads with high scores. The number of vehicles which choose to travel on a road at a junction is proportional to its score. More vehicles will travel on the roads with higher scores. The probability that a car will drive on route R_{xy} from junction j is set to the average travel time on that route to the power of 3 divided by the sum of all travel times of the routes leaving that junction to the power 3. This gives us the following equation.

$$Prob(R_{xy}) = \frac{(TIME\_R_{xy})^3}{\sum_i (TIME\_R_{ij})^3} \qquad (1)$$

TimeAnts extends Dijkstra lowest edge weight algorithm, with load balancing along congested edges, to determine the route. In other words the route with the lowest overall score will be chosen, as long as none of the edges chosen have a lower congestion score than the threshold. The congestion score is determined from the speed of vehicles on an edge divided by the speed limit of an edge. If there is a congested edge, vehicles will be distributed to other edges at this point leading to multiple good but sub-optimal routes for vehicles to follow.

A full graph search, from the origin to the destination, is performed. The fastest edges are chosen in this fashion, unless an edge is congested; in this case load balancing is performed near this edge. This graph search is described in Algorithm 1. The algorithm makes use of the following functions: *no_following()* – returns the number of edges following an edge; *traffic_congestion()* returns the traffic congestion expected on an edge; *traveltime()* returns the expected travel time on an edge; *assign_random()* returns an edge determined by equation 1 from the random number generated R, the travel times and the following edges.

Algorithm 1 TimeAnts Algorithm - Pseudo Code

```
//from Origin (O) to Destination (D)
route_list.clear()
//loop: through edges on map
for all edge[i] in Map
//if destination reached exit
if (edge[i] == D)
{
    exit
}
//return number of edges leading from this edge
int n = no_following(edge[i])
total_travel_time = 0
float edge_traveltime = max_traveltime
//loop through following edges
for all edges[j] following edge[i]
{
    total_travel_time = total_travel_time +
        (traveltime(edge[j])^3)
    //if this edge is faster than previous ones consider it as
    fast edge
    if (edge_traveltime > traveltime(edge[j]))
    {
        fast_edge = edge[j]
        edge_traveltime = traveltime(edge[j])
    }
}
//return traffic congestion on edge
TC = traffic_congestion(fast_edge)
//check if edge is congested with traffic
if (TC < threshold)
    route_list.add(fast_edge)
else
{
    //Load balance here according to equation 1
    R = rand() % (total_travel_time)
    Rand_edge = assign_random(R, total_traveltime,
        edge[i])
    route_list.add(rand_edge)
}
```

IV. SIMULATION-BASED TESTING

A. Simulation environment and scenario description

Modeling and simulations were performed on the iTETRIS[19] platform. iTETRIS uses in conjunction both the road traffic simulator SUMO [20] and the wireless network simulator NS-3 [21]. iTETRIS is specially designed to support VANET simulations, allowing for dynamic rerouting, and the use of the IEEE 802.11p protocol. iTETRIS is an open source simulator funded by the EU seventh framework programme.

A map similar to the one described in Wu et al. [8] and illustrated in Figure 3 was employed in these simulation-based testing. In this map there are 5 junctions (J1-J5), and each of the junctions is connected to the others by six road

segments (Road0-Road5). The road segment lengths vary between 500m and 2000m. The speed limits on these segments vary between 11 m/s (~40kmph) and 35 m/s (~125kmph). Each junction has a set of traffic lights.
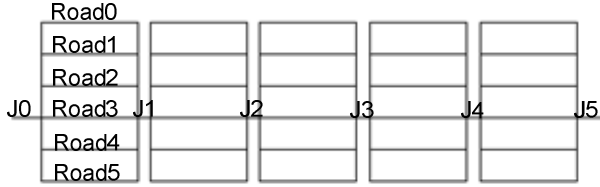

Figure 3 Map used for simulations

400 passenger cars are considered, which start driving into the map at time = 0; a new one appears on the map every second. If no rerouting mechanism is employed, the cars take the first turn at each junction. Time-Ants was deployed according to the description in the previous section. The congestion threshold was set to 80% for these simulations. The Time-Ants algorithm was tested against a mechanism which does not employ rerouting (the cars drive on the first turn on each junction – labeled None in Figure 4 and Table 2), Dijkstra lowest edge weight algorithm set to determine the fastest route (this will become congested if all the cars drive on it), the Dynamic rerouting algorithm (DNA) (described in the paper Wu et al. [8]) and an algorithm from Sommer et al.[22]

DNA is a VANET-based rerouting algorithm based on the fact that each vehicle communicates its speed and the distance to the car in front of it to all the other vehicles. This information is used to determine the edge weights for determining the best route using the Dijkstra algorithm.

DNA uses equation (2) to score each road segment, in which Sr_i is the score of the segment (graph edge), $\bar{S}_i$ reflects the average vehicle speed on the road segment). $\bar{D}_i$ - the inter-vehicle distance, $\bar{T}_i$ - the road type and $\bar{G}_i$ - the road segment length. k_j are weights.

$$Sr_i = k_1\bar{S}_i + k_2\bar{D}_i + k_3\bar{T}_i + k_4\bar{G}_i \quad (2)$$

Four DNA flavours are used in these tests, with the weights k1 through k4 indicated in Table 1.

Table 1 DNA flavor weights

Flavor	Weights
DNA1	1,0,0,0
DNA2	0,1,0,0
DNA3	0.5,0.5,0,0
DNA4	0,0,0,1

B. Testing Results

Figure 4 and Table 2 show the results for various iterations of Time-Ants when compared against different flavors of DNA, Dijkstra, the mechanism with no rerouting (None) and Sommer's algorithm.

The average travel time of a scheme can be lower, but fewer vehicles might get to their destination. So in order judge the effectiveness of the different schemes, the percentage of the vehicles which got to their destination within the time frame were recorded.

As can be seen from Figure 4, the proposed Time-Ants outperforms the other solutions in all simulations in terms of percentage of the vehicles which reach destination. For instance, the best performing DNA flavors, DNA1was outperformed by Time-Ants by 15% after the first iteration and 18% after the fourth iteration. Time-Ants also outperformed the mechanism which employs no rerouting by 16% after the first iteration and 19% after the fourth iteration.

The DNA4 algorithm performed poorly as it sent all the vehicles along the shortest route distance-wise, without regard to speed limits or traffic congestion.
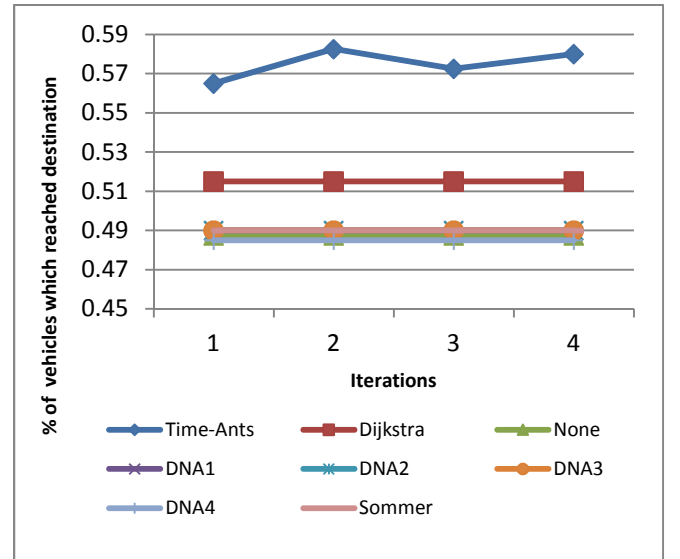

Figure 4 Percentage of vehicles which reached destination during simulation

Table 2 Percentage of vehicles to reach destination during simulation

Iterations	2	3	4	5
Dijkstra	0.515	0.515	0.515	0.515
None	0.4875	0.4875	0.4875	0.4875
Time-Ants	0.565	0.5825	0.5725	0.58
DNA1	0.49	0.49	0.49	0.49
DNA2	0.49	0.49	0.49	0.49
DNA3	0.49	0.49	0.49	0.49
DNA4	0.485	0.485	0.485	0.485
Sommer	0.49	0.49	0.49	0.49

The positive results of the Time-Ants algorithm are expected as, it considers load balancing, unlike the algorithm described in Wu et al. [8], for instance Hence Time-Ants makes better use of the full road network.

V. CONCLUSION AND FUTURE WORKS

This paper proposed Time-Ants a VANET-based ant-colony optimization algorithm which uses historical traffic data to improve traffic flows; this is a unique approach to dealing with traffic congestion. The simulation-based testing performed has compared Time-Ants against a number of other VANET-based routing techniques. Time-Ants outperformed the next best approach by 19% after four days of running the algorithm.

Future work will consider emissions as a parameter in a green extension of Time-Ants, which would minimize emissions instead of travel times. Also a more realistic vehicle trace will be used, such as the one from the TAPASCologne project [23], a 24 h vehicle trace of the city of Cologne, Germany. Simulation will also be run using vehicle counts from Dublin provided by Dublin City Council [24]. Note Time-Ants has to be updated to account for weekends which have different traffic patterns than week days.

REFERENCES

[1] V. Jain, A. Sharma, and L. Subramanian, "Road traffic congestion in the developing world," in *Proceedings of the 2nd ACM Symposium on Computing for Development*, 2012, p. 11.

[2] J. Austin, Brimblecombe, P, and Struges, W, *Air pollution Science for the 21st Century. Chapter 7 New directions: Air pollution and road traffic in developing countries.* .

[3] "WorldOMeters." [Online]. Available: http://www.worldometers.info/cars/.

[4] "TomTom." [Online]. Available: http://www.tomtom.com/en_ie/.

[5] "Google Maps." [Online]. Available: http://maps.google.ie/.

[6] H. Rakha and M. Van Aerde, "Statistical analysis of day-to-day variations in real-time traffic flow data," *Transportation research record*, pp. 26–34, 1995.

[7] L. H. Immers and S. Logghe, "Traffic flow theory," *Faculty of Engineering, Department of Civil Engineering, Section Traffic and Infrastructure, Kasteelpark Arenberg*, vol. 40, 2002.

[8] Y.-J. Wu and W.-C. Sung, "A dynamic navigation scheme for vehicular ad hoc networks," in *Networked Computing and Advanced Information Management (NCM), 2010 Sixth International Conference on*, 2010, pp. 231–235.

[9] S. Senge and H. F. Wedde, "2-Way evaluation of the distributed BeeJamA vehicle routing approach," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*, 2012, pp. 205–210.

[10] K. Collins and G.-M. Muntean, "A vehicle route management solution enabled by Wireless Vehicular Networks," in *INFOCOM Workshops 2008, IEEE*, 2008, pp. 1–6.

[11] R. Doolan and G.-M. Muntean, "VANET-enabled Eco-friendly Road Characteristics-aware Routing for Vehicular Traffic," in *IEEE Vehicular Technology Conference*, 2013, pp. 1–5.

[12] C. N. Chuah, H. Du, D. Ghosal, B. Khorashadi, B. Liu, C. Smith, and H. M. Zhang, "Distributed vehicular traffic control and safety applications with VGrid," in *Wireless Hive Networks Conference, 2008. WHNC 2008. IEEE*, 2008, pp. 1–5.

[13] M. Ferreira and P. M. d' Orey, "On the impact of virtual traffic lights on carbon emissions mitigation," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 1, pp. 284–295, 2012.

[14] C. Olaverri-Monreal, P. Gomes, R. Fernandes, F. Vieira, and M. Ferreira, "The See-Through System: A VANET-enabled assistant for overtaking maneuvers," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, 2010, pp. 123–128.

[15] D. Reichardt, M. Miglietta, L. Moretti, P. Morsink, and W. Schulz, "CarTALK 2000: Safe and comfortable driving based upon inter-vehicle-communication," in *Intelligent Vehicle Symposium, 2002. IEEE*, 2002, vol. 2, pp. 545–550.

[16] G. Beni, "From swarm intelligence to swarm robotics," in *Swarm Robotics*, Springer, 2005, pp. 1–9.

[17] D. Teodorović, "Swarm intelligence systems for transportation engineering: Principles and applications," *Transportation Research Part C: Emerging Technologies*, vol. 16, no. 6, pp. 651–667, Dec. 2008.

[18] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic algorithms: foundations and applications*, Springer, 2009, pp. 169–178.

[19] M. Rondinone, J. Maneros, D. Krajzewicz, R. Bauza, P. Cataldi, F. Hrizi, J. Gozalvez, V. Kumar, M. Röckl, and L. Lin, "iTETRIS: a modular simulation platform for the large scale evaluation of cooperative ITS applications," *Simulation Modelling Practice and Theory*, vol. 34, pp. 99–125, 2013.

[20] "SUMO." [Online]. Available: sumo.sourceforge.net.

[21] "Network simulator-3." [Online]. Available: www.nsnam.org.

[22] C. Sommer, R. Krul, R. German, and F. Dressler, "Emissions vs. travel time: simulative evaluation of the environmental impact of ITS," in *Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st*, 2010, pp. 1–5.

[23] "TAPASCologne." [Online]. Available: http://kolntrace.project.citi-lab.fr/.

[24] "Dublin City Council." [Online]. Available: http://www.dublinked.ie/datastore/datastore.php.