JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 39, 129-147 (2023)
DOI: 10.6688/JISE.202301_39(1).0008

Reinforcement Learning in Path Lifetime Routing Algorithm for VANETs

LINCOLN HERBERT TEIXEIRA AND ÁRPÁD HUSZÁK
Department of Networked Systems and Services
Budapest University of Technology and Economics
Budapest, H-1111 Hungary
E-mail: {teixeira; huszak}@hit.bme.hu

Continuous dynamic in vehicle movement creates a complex topology environment and connectivity relationship in Vehicular Ad-hoc Networks (VANETs). Consequently, overcoming longer transmission connections and reduced transmission reliability has been a significant problem. During the development of routing methods, fundamental research focuses on identifying and selecting paths with short distances or pulverization of information in environments with high traffic density to transmit packets successfully. As a result, transmission losses are minimized, and transport conditions are avoided; however, these techniques struggle to achieve more stable path routes in real-time environments because of sudden increases in traffic levels. Consequently, a model that effectively monitors the distance between vehicles and assists VANETs in selecting paths automatically and more accurately is proposed. To approximate and extend the lifetime of the communication path, we use an OpenAI Gym environment where a Reinforcement Learning (RL) agent can learn the route using the distance between cars as a model and define policies with higher lifetime rewards. The algorithm used a trained agent with the Proximal Policy Optimization (PPO) and Advanced Actor-Critic (A2C) algorithms in our environment. Applying machine learning on VANETs results in improved network efficiency, which offers real-time routing information in mobile environments. The simulation results show that RL-based routing extends the useful life of the route among the investigated methods between the origin and destination hosts.

Keywords: vehicular ad-hoc network, path lifetime routing, reinforcement learning, A2C, PPO

1. INTRODUCTION

Several new smart city concepts have been introduced in recent years, where vehicle communication plays an essential role in Intelligent Traffic Systems (ITS) [1]. Ad-hoc Vehicle Networks (VANETs) are wireless networks formed in real-time without additional network infrastructure. These in-car communication devices are used to communicate between vehicles, whether for traffic safety or entertainment. Unlike other ad-hoc devices, vehicular networks are not concerned with energy consumption for communication, as the expense is little compared to the energy spent for the vehicle to move. Each car can be either a source or a target node. If the origin unit is far from the destination vehicle,

Received September 14, 2021; revised November 4 & December 13, 2021; accepted December 18, 2021.
Communicated by Xiaohong Jiang.

packet transmission occurs using routes to reach the destination node [2]. That is, vehicles traveling between the origin and destination can become intermediate nodes. These ad-hoc networks can be established anytime, anywhere, allowing cars to communicate over a network in real-time and with great flexibility. Because of the dynamic topology of vehicles on the streets, it might result in various route choices, especially in a high-density environment [3]. Therefore, it is the function of the ad-hoc routing protocol to choose the communication route between the vehicles that best fit each type of communication. These algorithms allow distant hosts to communicate by forwarding messages through intermediate devices. The high efficiency and low latency of vehicle communications are essential than ever, and accurate delay management remains a significant problem in vehicles network [4, 5]. The deviation in vehicular networks is mobility. However, unlike other mobile networks, cars have limited mobility, as they have to follow the pattern of city streets and their speed limitations, traffic lights, and congestion jam. The exponential increase in network traffic volume and changing infrastructure needs require more intelligent routing approaches than in the past [6]. A routing algorithm that allows a longer lifetime in the communication between origin and destination is beneficial to optimize the functioning of the network. When one of these devices, which is part of the path chosen by the routing protocol, runs out of range of communication, the node becomes inactive, and the route is broken. As a result, in vehicular ad-hoc networks, maintaining the connection of each application is critical. In an ad-hoc vehicular network, there are two types of traditional algorithms for identifying a route. The first is based on a table, while the second is determined by demand.

A table-based routing protocol aims to maintain all nodes' routing information stable and up to date. Each device has a table to store the routing information and periodically shares this information with neighboring nodes to keep this information up to date. As the network topology changes, each node automatically updates its routing information and informs its neighbors. Destination Sequenced Distance Vector (DSDV) is a table-controlled protocol technique that chooses routes based on the minimum hopping requirement [7].

In the on-demand approach, the routing system is activated when the node has data to transmit. As a result, the overhead associated with managing routing information is minimized. When doing routing exploration, on the other hand, they must send packets. The productivity of a vast network topology is limited. Consequently, on-demand routing is typically applied to small environments. Two well-known on-demand routing protocols are Dynamic Source Routing (DSR) [8] and Ad-hoc On-Demand Distance Vector (AODV) [9]. The distinction between them is that in DSR, each node has a routing buffer where the most recent data is stored. If the current path fails, it would be simple to find a new path.

In this work, we evaluate and examine the effectiveness of two routing methods used in VANETs, DSR, and DSDV, and then compare them to our proposed RL-based approach. Since they are a commonly used form of ad-hoc communication in vehicular networks today, as traffic congestion worsens, routing algorithms for ad-hoc vehicle networks must be optimized. Even a millisecond flaw in the traffic control network can have catastrophic consequences. Therefore, in our work, a routing protocol technique on reinforcement learning is proposed to increase the useful life of the route, minimizing the number of reconnections to a common destination. This study intends to suggest a solution that considers the distance between each host and uses the location of vehicles as a parameter, allowing each host to serve as a node in the network and exchange information quickly and efficiently. As a result, based on the location of the destination car, an algorithm using reinforcement learning is proposed to help identify the route with the most extended lifespan rather than the shortest path, as most existing routing protocols do.

Methods

We provide strategies for enhancing the performance of our suggested technique, such as (1) using knowledge from adjacent node locations to choose the best path and avoiding hosts that do not bind to the target host to prevent packet loops; (2) choosing the route with the most extended lifespan witch uses a reward feature that calculates the distance among the hosts, and (3) using performance parameters such as route lifetime, amount of reconnections, and hop counts, we measure the efficiency of the suggested method. The key contributions of this article are:

- We create a reinforcement learning environment where an agent can learn to find the path with the most extended lifetime. A procedure determines the chosen path based on the distance between the nodes, and as a reward, the path's lifespan is connected.
- The impacts of existing procedures were compared between traditional algorithms to the novel paradigm suggested in this work utilizing computer-based simulation tools.

Although reinforcement learning has significantly influenced other fields, we believe its promise in networking has not been completely realized. By providing an OpenAI Gym setting, we hope to encourage studying the solutions using reinforcement learning in routing networks. The remaining of this research paper is organized as follows: Section 2 examines the associated studies on network path protocol. Section 3 explains the proposed algorithm. Then, the simulation setting is defined in Section 4. In Section 5, we go into detail about how to evaluate the simulation performance. Finally, the paper comes to a close with discussions in Section 6.

2. RELATED WORK

As related works, we will approach routing protocols in VANETs, being two traditional protocols widely used in ad-hoc networks, DSR and DSDV. These protocols were used as a comparison in the performance of the proposed new solutions. Then, solutions related to the lifetime of the routing path will be presented and, finally, how is the research nowadays associated with machine learning applied to VANET.

2.1 Routing Protocols in VANETS

Ad-hoc networks are made up of wireless nodes with no centralized system, and their topology and connection capabilities change frequently. These networks are classified as MANETs (Mobile Ad-hoc Networks) when node movement is taken into account [10]. Vehicle Networks (VANETs) are a form of MANET in which the nodes are vehicles that circulate in cities or on highways. Cars in motion in current and potential Intelligent Transport Systems would collect real-time data regarding traffic using sensors installed around the streets. VANETs improve road protection by allowing automobiles to share information about collisions, traffic congestion, barriers, road distortion, and traffic jams. The mobility of VANETs is an essential factor to remember when it comes to routing. When transmitting safety messages between vehicles, the main issues to consider are message exchange delay constraints and complex topology variations [11].

The conventional routing strategy considers the network a weighted graph and chooses the route with the minimum cost, satisfying QoS criteria [12]. Several relevant indicators, such as latency, reliability, stability, and resource usage, are included in the challenge to be solved. When several metrics are needed, the routing problem becomes non-deterministic polynomial-time complete [13]. Heuristics have been used to achieve near-optimal results. Research in route protocols is extended and has explored numerous routing methods applied to different network problems. Most modern networks are complex, and route optimization when accounting for complex topology variations is a significant issue in routing. Ad-hoc vehicular networks, for example, have nodes that move, resulting in topology changes. As a result, this topology changes causes break link connection in some cases, causing the operation between nodes to be disrupted. Traditional routing strategies based on significant assumptions about traffic patterns and network status improvements are increasingly seen as impractical in dealing with the dynamic and rapidly changing mobile environments. If the basic premises are not met in real-time, the network output can deviate significantly from the expected. Due to this, reinforcement learning is a viable solution for dealing with variable network conditions.

There were also different protocols established for transmitting traffic data without the usage of the roadside unit. Based on the density and location of the car, this technique improves routing positively and attentively and decreases congestion along major thoroughfares while effectively maintaining traffic reports [14].

A modern simplified multistage p-persistent protocol for vehicle-to-vehicle communications has been proposed to handle the associated explicit stochastic delay and flow control capacities. The geometric distribution would reach the upper limit of the stochastic uncertainty in the basic scenario where nodes join the pipe. According to the simulation findings, the theoretical delay model would provide an overview of the upper limit delay for services in vehicular networks [15]. An empirical technique to modeling expected latency was created to tackle the problem and enhance the efficiency of the lower limit, utilizing a V2V connection selection algorithm focused on greedy cells [16].

The Vehicle-to-everything (V2X) architecture was developed to estimate the intensity of road traffic [17]. This system uses sensors to collect details from RSUs and cars, and the topology of the roadmap is modeled to precisely measure the vehicle's instantaneous density. The V2X methods broaden the coverage area and improve the precision of the obtained data, increasing the robustness and fault tolerance of the technique. Because it is aware of the instantaneous density of cars, this system defines distribution protocols for the control of traffic flow mitigation measures [18]. In addition, a control algorithm that takes V2V interactions and vehicle details into account for equal action and reduced waiting time has been developed [19].

Now, we will look at the two conventional ad-hoc network protocols that were used in this research. We choose these two routing protocols to address different types in our simulation: reactive (DSR) and proactive (DSDV).

DSR: Dynamic source routing protocol is a straightforward, reliable routing protocol intended for ad-hoc wireless networks with several hops of moving nodes. DSR enables the network system to be fully organized and optimized, with no prior network infrastructure or management required [20].

The protocol comprises two frameworks, discovery and maintenance, that operate together to enable the nodes on the ad-hoc network to learn and sustain routes between arbitrary destinations. Using source routing eliminates the need for routing details on the intermediary hosts from which information is redirected and permits hosts to deliver and store this knowledge for future use. Both features of the algorithm run on-demand, resulting in a DSR routing packet header that immediately scales to just what is required to adapt to changes in the routing table in use [8].

DSDV: The proactive routing protocol of Destination-Sequenced Distance Vector uses the Bellman-Ford algorithm to build a routing table. DSDV creates a routing table for each network node. The routing algorithm DSDV made a significant contribution by solving the routing loop problem. Because each node manages the routing table and this table needs to be updated more frequently if a router receives new information. With this information in the routing table, the source sends the packet to the destination via intermediate nodes [21]. Each node must identify the next hop to the collector and the number of stages away from the collector. This information can be found in the routing table [22]. The routing table entry for each node provides information such as the IP address of the node's most recent known sequence number and the hop count utilized to reach that node. The database often contains a record of the next-hop neighbor to join the destination node and the timestamp of the most recent update received for that node [23]. The packet distribution rate in DSDV decreases sharply, and one of the critical explanations for this is the usage of routes that have already been closed due to broken connections. The presence of disrupted pathways in the DSDV does not rule out the possibility of identifying the direction to the destination [24]. However, as automobiles increase compared to the other protocols, the finding shows that the DSDV protocol negatively impacts. DSDV cannot be considered as a viable choice either if the number of vehicles rises significantly [22].

The path is chosen by DSR and DSDV depending on the number of leaps. Even though the selected route contains the lowest hops, the nodes may be far apart. As a result, each node expends more resources on data transmission, increasing the likelihood that an intermediate link will fail, necessitating a new route request [2]. Because of this open issue, we propose a route-choice model that uses the path's lifetime as a metric rather than the number of hops.

2.2 Path Lifetime

The majority of research has been focused on developing routing algorithms that extend the route's existence. The researchers, however, take energy usage into account and attempt to extend the battery life of ad-hoc network devices where power is a limiting factor [25].

A cluster-based data collection approach to increase power quality and meet delay constraints has been developed for ad-hoc networks [26]. In the first step, the sensors are grouped into clusters to ensure that they can interact with the mobile data collector within predefined hops. The route was then established using a genetic algorithm, an applicable technique for the optimization problem concerning discovering the shortest path. Comparisons of the path's useful lifetime have been made [2]. Researchers assumed the nodes begin with varying degrees of control, and when a node in a segment dies, the course is interrupted. The lifetime of each node in a route was calculated, and the shortest was chosen to be the path's lifetime, based on the total energy available at the node. The concept is compatible with the idea that its weakest node defines a path's strength. They evaluate the route lifetimes obtained by different simulation routing algorithms. The transmitting latency has been ignored for convenience, and the time required to react to control signals has been calculated. The simulation findings show that the Maximum Path Lifetime Routing (MPLR) [27] outperforms other models in route lifespan. In these instances, as the scale of the region grows, life expectancy decreases dramatically.

Although the concepts are relevant to improving the valuable lifetime of the path, such as avoiding the weakest node, since energy constraints do not apply to the vehicle network requirements, we did not use these protocols in this work as a reference in the simulations.

2.3 Machine Learning Applied in VANETs

Because of the nature of VANETs, there are several variables, such as disconnections, multiple paths, vehicle movements, sometimes corresponding to the layout of city routes, with varying speed limits, and parking or stuck in traffic. Using modern technology, such as Machine Learning (ML), makes sense to detect these trends and create more effective communication routes as an alternative to conventional network protocols [28].

ML network applications aim to automatically learn system dynamics, such as new traffic entries, bottleneck sites, architecture enhancements, connection efficiency, and power usage, to improve the Quality of service provided to end-users while maximizing network capital and supplier revenues [11].

Reinforcement learning (RL) is a machine learning class that offers a system to benefit from past experiences with its environment to pick its strategy reliably. At the same time, the agent has no previous knowledge about the system environment [29]. RL has developed autonomous systems that grow with expertise in various areas, including games [30], health [31], robotics, power, networks, and telecommunications. It is widely recognized that RL is appropriate for addressing optimization difficulties in distributed systems in general, such as network routing. RL has a moderate overhead in control packages, memory, and computational power than other optimization approaches used to tackle the same issues. Many protocols have been suggested, each contributing significantly or marginally to the area of optimum route selection for packet transmission over different kinds of communication networks under differing QoS circumstances [11]. Specifically, the agent examines the present condition before acting and receiving the compensation along with the new state. The observed detail, namely the reward and the new state, is used to modify the agent's policy, which is then replicated until the policy approaches the optimum policy.

Several researchers have proposed introducing quality metrics such as delay, jitter, and loss in the routing algorithm. Machine learning approaches such as Q-learning and neural networks were utilized to compute the candidate route for sending packets [32, 33]. A concept of a cognitive routing network was established to drive state-space optimization in a complex context [34]. Deep Reinforcement Learning (DRL)-based approaches have been used by researchers to assist cognitive routing research [6, 35].

Another important aspect of previous research is that vehicle positioning information is shared between vehicles or a central controller via routing protocols used in vehicular networks. As a result, there is a significant amount of control overhead generated. Furthermore, the sudden shift in automobiles necessitates more time for integration, which allows the vehicle to receive incorrect details in real-time and influences the routing protocol's behavior. As there is a hop measurement in the routing protocol, the cars collect information only from their subsequent road section, contributing to the optimum local challenge. As a result of these considerations, machine learning models may aid in route discovery on VANETs [3]. The machine learning algorithm estimates the situation based on previous and current real-time vehicle states. This prediction serves as input to the system, allowing it to route data efficiently.

Based on the positioning of the vehicles, the analysis of the RL model utilized in this study tracks, analyzes and predicts the action of the communication route choice. In this study, the PPO and A2C algorithms were used, which are already used in neural networks and reinforcement learning. These algorithms were chosen for their working characteristics in our environment, which requires discrete parameters (vehicle numbers) and multiple arrays of discrete values in the observation space (vehicles positioning).

A2C: When constructing machine learning algorithms, two costs must be considered: sample complexity and computing complexity. The number of interface stages between the agent and its environment is referred to as the sample complexity. On the other hand, computational complexity refers to the number of numerical operations that must be performed. The Asynchronous Advantage Actor-Critic (A2C) is a deterministic version of the Asynchronous Advantage Actor-Critic (A3C) that performs equally well. The method contains several fundamental notions, such as an updating system that calculates the advantage function using fixed-length trial segments and shared layer structures between regulation and the meaning resource. The researchers discovered an alternative to asynchronous implementation, where a deterministic execution expects an actor to complete the experience before executing an update that averages all actors. This approach has the benefit of making better use of GPUs, which perform well for big batches. This algorithm is known as A2C, which stands for an actor-critic of advantage. The synchronous A2C implementation outperforms the asynchronous deployments; however, there is no indication that the noise caused by asynchrony improves efficiency. When utilizing single GPU computers, the A2C performance is more cost-effective than A3C and quicker than one CPU-only A3C implementation when using broader policies [36].

PPO: With supervised learning, we can quickly enforce the cost function, apply a gradient decrease to it, and be sure that we can get excellent results with minimal hyperparameter modification. The road to performance reinforcement learning is not so clear because the algorithms have several moving parts challenging to debug and take a significant amount of improvement effort to achieve successful implementation. The Proximal Policy Optimization (PPO) algorithm attempts to compromise between simplicity of execution, sample sophistication, and ease of modification, calculating an update for each point that minimizes the cost function while ensuring that the variance from the previous policy is comparatively minimal. A PPO version in which an adaptive KL penalty regulates the policy adjustment in each iteration. The latest version employs a novel objective function not contained in other algorithms:

$$L^{CLIP}(\theta) = \hat{E}_t[min(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)]. \quad (1)$$

This goal applies a method for making a trusted area update consistent with Stochastic Gradient Descent. It simplifies the algorithm by eliminating the KL penalty and the need for adaptive updates. Despite becoming somewhat easier to execute, this algorithm performed the highest in continuous control tasks in simulations [37].

3. RL-BASED PATH LIFETIME ROUTING APPROACH

This paper proposes a routing method that considers the distance between nodes and the route's lifetime. Path lifetime is the amount of time predicted until one or more nodes in the path leave the signal range, rendering the route invalid. Traditional ad hoc routing systems, such as DSDV and DSR, choose a route depending on the number of hops. Even though the selected path contains a limited number of intermediate nodes between the origin and the destination, the distance between each pair is often high. If intermediate nodes with shorter distances between them are selected instead, the chance of the path's longevity rises.

Through reward and punishment, reinforcement learning models evaluate the action taken at each level of the system that changes state; the agent learns to maximize performance. States reflect the number of vehicles that are part of the same communication segment, such as the city center or a specific neighborhood, in the proposed model. Actions indicate the decision of the route the vehicles will take to communicate — in other words, the path between origin and destination. The reward is estimated as the lifetime of the chosen path after specifying numerous actions together since the system's goal is to maximize communication time to avoid reconnections.

3.1 Environment

OpenAI Gym [38] is an open-source framework for training, testing, and benchmarking reinforcement learning algorithms. It includes a variety of activities, such as those used in arcade games. Here we describe how the platform might be used as a simulation, test, and diagnostic paradigm for networking search graph conditions. We use the OpenAI Gym tool to create the $PathLifetime$ environment that contains the following components:

- The state-space S, consisting of the current state of the graph (in terms of a multi-dimensional output of a process-based pairwise distance model) and a multifaceted observation;
- The action space A, which consists of the next hop of the graph;
- Transitions between states, governed by the deterministic process-based model and the distance sequence;
- The reward r, which encourages the lifetime of the path.

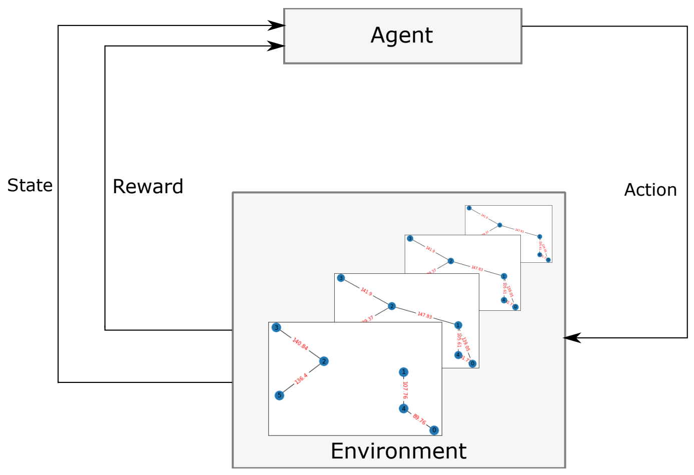

Fig. 1. Reinforcement learning model: Interaction between the agent and the gym environment.

The information flow is shown schematically in Fig. 1. The framework can be applied to any process-based finding path model. In the following sections, we detail our implementation choices.

Observation space: As shown in Fig. 1, the observation state is composed of the current graph and its three previous states. Since we cannot know which way the cars are moving, this violates Markov's property [39]. The alternative is to hold some observations from the past and use them as a state. We propose keeping four successive locations together and observing each state; thus, this preprocessing stacks four rows, resulting in final state space in the matrix of $4 \times N$ columns. In other words, it is composed of a matrix that has the combination of all possible communication pairs (links) based on the distance between the vehicles. The matrix is exemplified in Eq. (2).

$$S = \begin{bmatrix} D_t(0,1) & \dots & D_t((n-1),n) \\ D_{t-1}(0,1) & \dots & D_{t-1}((n-1),n) \\ D_{t-2}(0,1) & \dots & D_{t-2}((n-1),n) \\ D_{t-3}(0,1) & \dots & D_{t-3}((n-1),n) \end{bmatrix} \tag{2}$$

Where D means the distance between the hosts, t is the current timestamp, n is the number of vehicles in a given area, and the number of columns of the matrix is the combination of the pairs of hosts without repetition given by Eq. (3) as known as the binomial coefficient of pairs.

$$\binom{n}{2} = \frac{n!}{2!(n-2)!} \tag{3}$$

Action space: The action space is discrete in this case and given the number of vehicles in a given area, such as the downtown. It consists of cars that can be part of the path, either as origin, destination, or intermediate nodes.

Reward: The reward function is defined in three zones. The success zone corresponds to the destination of the package. The viability zone is determined when the chosen S state can be part of the path. The rupture zone is when the selected state does not exist or cannot be part of the path (due to the distance between the current node and the next hop), but the reward function will receive the reward value if it reaches the success zone. Therefore, we can summarize the reward as Eq. (4).

$$R = \begin{cases} lifetime & ; if\ t = dst \\ 0 & ; otherwise \end{cases} \tag{4}$$

Where t means the target, dst the destination node. We also use the path search condition and route selection function described in Eq. (5).

$$C = \begin{cases} stop & ; if\ t = d \\ nextaction & ; if\ |E| \exists \\ searchnode & ; if\ |E| \nexists \\ avoidloop & ; if\ s \in Path \end{cases} \tag{5}$$

Where d is the distance, and r is the range of communication in meters.

3.2 Algorithm

The formalization of the method:

Let $H < V, E, D >$ a direct graph with an attribute related to the given VANET architecture, where communication may take place in both directions. Where $V = \{n_0, ..., n_W\}$ is the set of vertices with $|V| = N$, where N is the number of nodes, in this case, the vehicles that can participate in the communication, and $E = \{(n_0, n_1)_1, (n1, n_m)_2, ..., (n_j, n_D)_M\}$ is the set of edges with $|E| = M$; and D is a set of non-negative attribute functions. For each edge $e = (u, v) \in E$, there is a function: weight function $d_e(t) \in D$ where d is the distance variable between the two nodes. A weight function $d_e(t)$ specifies the distances from u to v, starting from u in time t.

Algorithm 1 represents the pseudo-code used to describe the step function of the *PathLifetime* OpenAi Gym environment. The reinforcement learning algorithms will learn to find the longest lifetime path of the dynamic graphs.

We have four main functions in the algorithm: *Lifetime*, *Reset*, *Observation*, and *Step*. The *Lifetime* function calculates the time of life of the chosen path. The *Reset* function is called when the path loops. The *Observation* function calculates the distance between all vehicles based on the Cartesian position of each vehicle and stores the current length and the last four distances between cars. The primary function of the algorithm is the *Step* function, where reinforcement learning algorithms will learn after each action to seek the greatest reward, that is, an extended lifetime of the chosen path, based on the observation space. The *Step* function has four test conditions. The first is whether there is a link between the current position and the next node chosen by the action. It proceeds to the following tests; otherwise, the algorithm must select another node to be part of the route. We need the test condition $action \in path$ to avoid looping; if the chosen action is already part of the path, it means that a specific node has already been selected previously

Algorithm 1 : Step Function

```
if $H.has\_edge(position, action)$ then
    if $action = target$ : then
        $reward = lifetime()$
        $done = True$
    else
        if $action \in path$ : then
            $reward = 0$
            $done = False$
            $reset()$
        else
            $path.append(action)$
            $position = action$
            $reward = 0$
            $done = False$
        end if
    end if
end if
```

to be part of the path to close a cycle. It means that the route enters a loop, which is not desirable in any routing protocol. That is why the *Reset* function is called. After the first condition, it will be tested whether the chosen action is equal to the target; if so, it means that the path has reached its final destination, and it is possible to calculate the lifetime of that path and finish the execution of the algorithm for that particular path. If the chosen action is not part of the path yet, this is a suitable action and can be selected. However, the reward is still null, and the algorithm is not finished. Just one more node was found that could be part of the route path. At the end of the condition tests, the Observation function returns the four standard variables of the OpenAi Gym environment (*obs*, *reward*, *done*, $info$).

The *NetworkX* tool is a package of the Python programming language, which creates and manipulates the dynamic structure of complex networks [40]. This tool was used to build the connection graph between vehicles. The two main functions, $H.has\_edge$, are to check if the link between the current property and the next node is still active, and $H.has\_path$ to check how long the route remains active, making it possible to calculate the route lifetime.

4. SIMULATED ENVIRONMENT

Researchers often use computer simulation programs to test and assess the findings of their study since they are more flexible and less expensive than real-world implementation. To assess the performance of the suggested approach, we ran a computer-based simulation. OpenAI Gym and Python were used as simulation tools because both are computationally efficient and practical. A generic grid map of the city was used as a backdrop in our simulation, called the Manhattan [41] grid. We decided on a time limit of five minutes for this simulation activity, and the Urban Mobility (SUMO) [42] simulation was used to create a realistic environment. The SUMO program is open-source and offers a simple framework for simulating land transport modes. It generates a simulation in which automobiles and pedestrians can be viewed as nodes, and a network is constructed. It includes a collection of improvised tools that serve as a valuable platform for establishing numerous situations that a researcher can consider for his experiment [24].

Fig. 2 displays a city with a five-block grid layout. This model uses a grid road design to mimic the movement pattern of mobile nodes in metropolitan situations. This mobility model features horizontal and vertical highways, nodes put on the map at random at the start of the simulation, and the option to change lanes. When a node hits a junction, it may move left, right or straight, randomly [43].

Fig. 3 illustrates the Diagram Process Flow for the Experimental Setup, and various stages are required to build up this model. First, we created the map using the *Netedit* program (Manhattan grid). The *randomTrips* program was used to produce paths of the cars, resulting in the *route.xml*. The configuration file *sumo.cfg* was used to complete the first phase of the simulation with the SUMO tool. The *traceExporter* program may generate the *gpsDat* file used in the python simulation of the proposed OpenAI Gym algorithm and the two other routing algorithm implementations utilized in today's VANETs, the DSR and DSDV routing protocols.

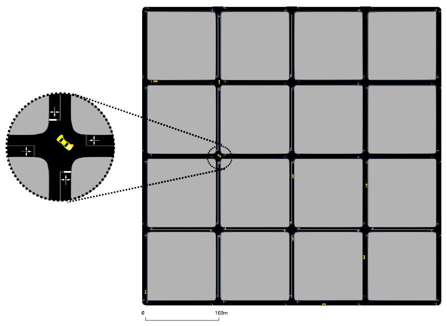

Fig. 2. The urban environment in SUMO is configured with a five block topology simulation. This model portrays a typical urban area.

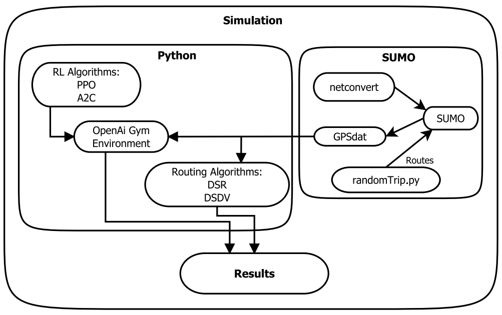

Fig. 3. The information flow utilized to illustrate and test the proposed method is depicted in the simulation configuration diagram.

5. RESULTS AND DISCUSSION

Simulations were carried out to show the efficiency of the proposed algorithm. Comparison between the two existing protocols, DSR and DSDV, and our proposed scheme that uses reinforcement learning was made. Each protocol was simulated with ten vehicles being evenly distributed and random travel to generate traffic, as explained in the simulation chapter. Three different communication ranges, with 100, 200, and 300 meters, were also used to test the efficiency of the proposed algorithms.

The following metrics were used to evaluate the performance of the proposed protocol: (1) $Lifetime$ mean of the path, that is, the path that exists between the same source and destination. (2) Number of connection transitions, that is, the number of route changes between the source-destination path during the simulation time. (3) The $LengthPath$ mean is the average number of hops that routes have. As shown in Fig. 4, PPO and A2C agents obtain a higher average reward than standard DSR and DSDV communication protocols. In other words, the route paths chosen by reinforcement learning agents have a longer life span. In the best scenario, when the maximum allowed speed is 30km/h and the communication range is 300 meters, the difference between the path chosen by the PPO agent compared to DSR or DSDV algorithms almost doubles the lifetime of the route path on average, during the simulation time.

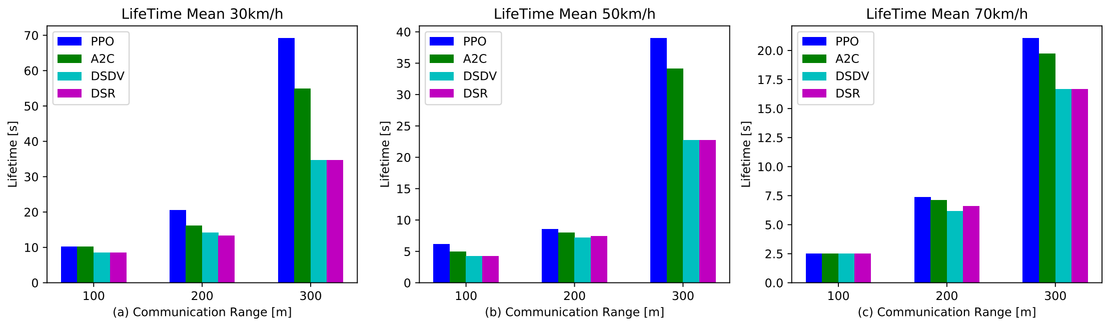

Fig. 4. Lifetime mean.

This evaluation is critical in data communication, especially in mobile vehicular networks, increasing the route's lifetime, consequently decreasing the number of reconnections and overheads of the protocol, as shown in Fig. 5.

We achieved half of the connection transitions with the proposed methodology than PPO and DSR or DSDV when we have a 30km/h speed and a communication range of 300 meters Fig. 5 (a). In the simulation time of five minutes, when we use the protocols of conventional networks, ten transitions occur. In contrast, when we use intelligent algorithms to increase the life of the route, the PPO algorithm occurs only five shifts of the route path. Although both artificial intelligence agents are superior to conventional protocols in all the simulations carried out, the PPO agent had a slight advantage compared to the A2C agent, either in the lifetime of the path or in the number of transitions.

As the lifetime of the chosen route paths is significantly longer, this does not mean that the selected route is the shortest; that is, it has fewer hops.

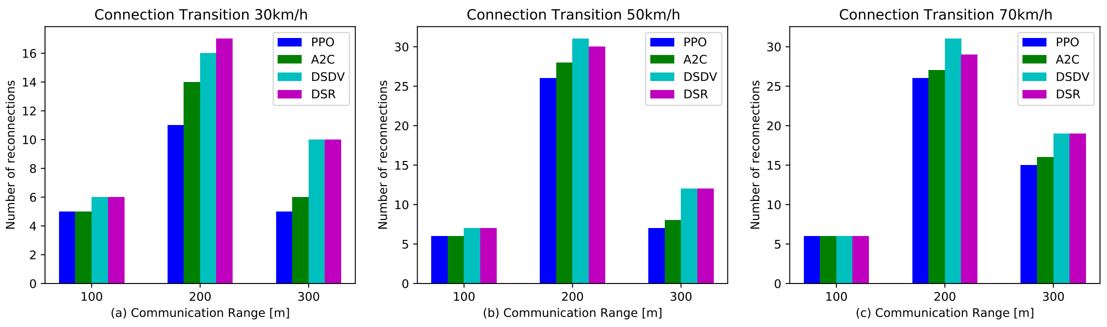

Fig. 5. Connection transition.

Next, the number of hops of the paths chosen by each algorithm obtained in the simulations will be compared. This information is essential, as each additional hop used in communication increases the probability of packet loss and signal attenuation.

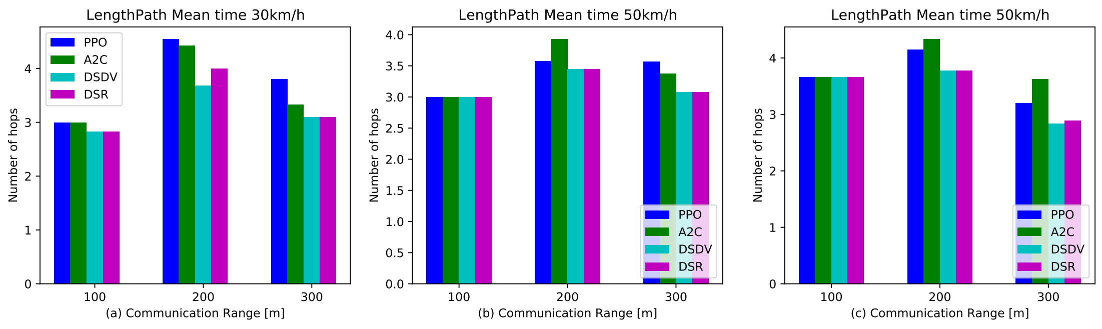

Fig. 6. LengthPath mean.

Therefore, as seen in Fig. 6, the average hop size of the paths chosen by intelligent agents is equal to or greater than traditional protocols, as expected. It happens because, in this work, we are not concerned with the number of hops on the chosen path but with the probability that it will live longer. However, although the average number of hops of the algorithms used in RL is higher than the traditional algorithms, DSDV and DSR, the difference is not significantly more prominent, reaching 23% in the worst case, seen in Fig. 6 (a). It means an average of two hops more compared to traditional algorithms. Two other figures were used with the same metric but exposed differently to visualize the lifetime gain compared to the path length (hops).

Fig. 7 shows the probability mass function of the measured path length in hops. This graph compares the route selection algorithms at three different speed limits of the cars, with 30, 50, and 70 km per hour, where 200 meters of radio range is considered for the vehicles to communicate.

On the other hand, Fig. 8 has the same simulation configuration and compares the reward, that is, the path lifetime of the chosen route, for each simulated algorithm. We can say that the paths chosen by the RL tool have a uniform distribution compared to traditional algorithms. We realize that traditional algorithms focus their choices on the shortest-lived paths because these algorithms are not concerned with maintaining connectivity, only with finding the shortest path. It further affirms the relevance of our work, as traditional protocols should not be used in highly mobile networks such as vehicular networks, but rather tools that help us choose efficient routes should be used nowadays.

In order to prove the effectiveness of the RL algorithms chosen to solve the problem proposed in this work, Fig. 9 shows the learning curve during model training, of the two algorithms used in this article, PPO and A2C. This graph shows us the average rewards accumulated during model training.

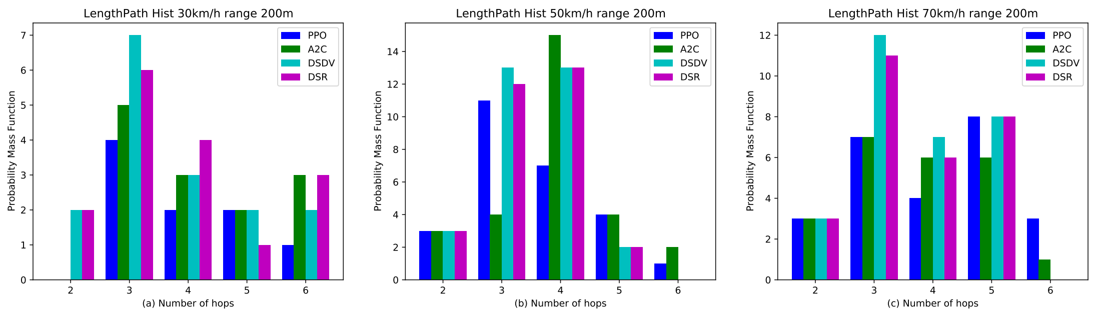

Fig. 7. LengthPath histogram.

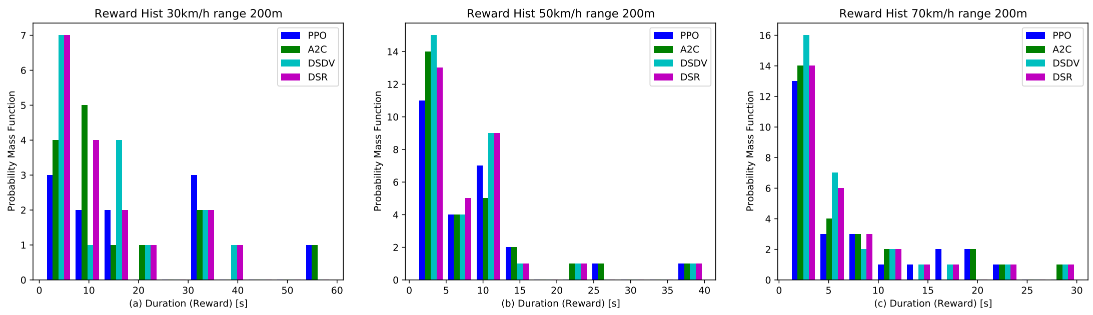

Fig. 8. Reward histogram.

6. CONCLUSION

We introduced a new Gym environment to study the routing network strategies for VANET and discussed the considerations we made during the design process of the *PathLifetime* Gym environment. We have implemented a reactive agent, standard practi-

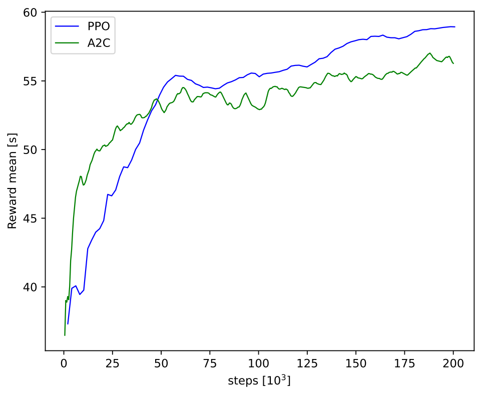

Fig. 9. Trainning model reward mean.

ce, and an agent trained with PPO and A2C that serve as a baseline for future investigations and research. The PPO agent learned strategies with a greater reward than the other chosen agent, A2C, demonstrating that reinforcement learning agents can identify different route paths for each chosen scenario and configuration, primarily if they used various forms of reward. Compared with other classic algorithms currently used as DSDC and DSR, intelligent learning algorithms, which used reinforcement learning and were presented in this work, significantly gained the studied metrics. The future of telecommunications applied to complex networks with high mobility and specific requirements, like VANETs, should not be tied to the classic route algorithms but should broaden their horizons. Using new machine learning technologies techniques to adapt to the future of data communication is an alternative.

REFERENCES

1. G. S. Khekare and A. V. Sakhare, “A smart city framework for intelligent traffic system using vanet,” in *Proceedings of IEEE International Mutli-Conference on Automation, Computing, Communication, Control and Compressed Sensing*, 2013, pp. 302-305.
2. H.-C. Wang and W.-H. Chen, “Maximum path lifetime routing for ad hoc wireless networks,” in *Proceedings of IEEE 9th IFIP International Conference on Mobile Wireless Communications Networks*, 2007, pp. 166-170.
3. M. Saravanan and P. Ganeshkumar, “Routing using reinforcement learning in vehicular ad hoc networks,” *Computational Intelligence*, Vol. 36, 2020, pp. 682-697.
4. V. Nguyen, T. T. Khanh, X.-Q. Pham, G.-W. Lee, and E.-N. Huh, “Performance analysis of adaptive mac protocol in vanets considering the potential impact on throughput and transmission delays,” *International Journal of Communication Systems*, Vol. 33, 2020, p. e4172.
5. Y. Xu, J. Liu, Y. Shen, J. Liu, X. Jiang, and T. Taleb, “Incentive jamming-based secure routing in decentralized internet of things,” *IEEE Internet of Things Journal*, Vol. 8, 2020, pp. 3000-3013.

6. J. Wu, J. Li, Y. Xiao, and J. Liu, “Towards cognitive routing based on deep reinforcement learning,” *arXiv Preprint*, 2020, arXiv:2003.12439.
7. L. Barolli, F. Amato, F. Moscato, T. Enokido, and M. Takizawa, *Web, Artificial Intelligence and Network Applications*, Springer Nature, Switzerland, Vol. 1150, 2020.
8. D. B. Johnson, D. A. Maltz, J. Broch *et al.*, “Dsr: The dynamic source routing protocol for multi-hop wireless ad hoc networks,” *Ad Hoc Networking*, Vol. 5, 2001, pp. 139-172.
9. I. Chakeres and E. Belding-Royer, “Aodv routing protocol implementation design,” in *Proceedings of the 24th International Conference on Distributed Computing Systems Workshops*, 2004, pp. 698-703.
10. S. Basagni, M. Conti, S. Giordano, and I. Stojmenovic, *Mobile Ad Hoc Networking*, John Wiley & Sons, 2004.
11. Z. Mammeri, “Reinforcement learning based routing in networks: Review and classification of approaches,” *IEEE Access*, Vol. 7, 2019, pp. 55916-55950.
12. Y. Xu, J. Liu, Y. Shen, X. Jiang, Y. Ji, and N. Shiratori, “Qos-aware secure routing design for wireless networks with selfish jammers,” *IEEE Transactions on Wireless Communications*, Vol. 20, 2021, pp. 4902-4916.
13. Z. Wang and J. Crowcroft, “Quality-of-service routing for supporting multimedia applications,” *IEEE Journal on Selected Areas in Communications*, Vol. 14, 1996, pp. 1228-1234.
14. G. Martuscelli, A. Boukerche, L. Foschini, and P. Bellavista, “V2v protocols for traffic congestion discovery along routes of interest in vanets: a quantitative study,” *Wireless Communications and Mobile Computing*, Vol. 16, 2016, pp. 2907-2923.
15. X. Chen, S. Leng, S. Mao, and F. Wu, “A generalized multi-stage p-persistent mac protocol for v2v communications,” in *Proceedings of IEEE 18th International Conference on Communication Technology*, 2018, pp. 488-493.
16. F. Abbas, P. Fan, and Z. Khan, “A novel low-latency v2v resource allocation scheme based on cellular v2x communications,” *IEEE Transactions on Intelligent Transportation Systems*, Vol. 20, 2018, pp. 2185-2197.
17. J. A. Sanguesa, J. Barrachina, M. Fogue, P. Garrido, F. J. Martinez, J.-C. Cano, C. T. Calafate, and P. Manzoni, “Sensing traffic density combining v2v and v2i wireless communications,” *Sensors*, Vol. 15, 2015, pp. 31794-31810.
18. M. Cheong, Y. Lee, W. Park, and I. Yeom, “Analysis of v2v communication for adas,” in *Proceedings of IEEE 9th International Conference on Ubiquitous and Future Networks*, 2017, pp. 284-286.
19. I. Pisa, G. Boquet, J. L. Vicario, A. Morell, and J. Serrano, “Vaima: A v2v based intersection traffic management algorithm,” in *Proceedings of IEEE 14th Annual Conference on Wireless On-demand Network Systems and Services*, 2018, pp. 125-128.
20. A. Shaheen, A. Gaamel, and A. Bahaj, “Comparison and analysis study between aodv dsr routing protocols in vanet with ieee 802.11,” *Journal of Ubiquitous Systems and Pervasive Networks*, Vol. 7, 2016, pp. 7-12.
21. M. Manjunath and D. Manjaiah, “Spatial dsdv (s-dsdv) routing algorithm for mobile ad hoc network,” in *Proceedings of IEEE International Conference on Contemporary Computing and Informatics*, 2014, pp. 625-629.

22. M. Abdelgadir, R. A. Saeed, and A. Babiker, "Mobility routing model for vehicular ad-hoc networks (vanets), smart city scenarios," *Vehicular Communications*, Vol. 9, 2017, pp. 154-161.
23. T. Mantoro and M. Reza, "Performance analysis of aodv and dsdv using sumo, move and ns2," in *Proceedings of IEEE International Conference on Informatics and Computing*, 2016, pp. 372-376.
24. V. Kumar and R. Kumar, "Smart re-route routing protocol for vehicular ad-hoc networks," in *Proceedings of IEEE Second International Conference on Inventive Research in Computing Applications*, 2020, pp. 758-763.
25. L. Romdhani and C. Bonnet, "Energy-based routing optimization in manet: Cross-layer benefits," in *Proceedings of IEEE 9th IFIP International Conference on Mobile Wireless Communications Networks*, 2007, pp. 96-100.
26. Q. Wu, P. Sun, and A. Boukerche, "A novel data collector path optimization method for lifetime prolonging in wireless sensor networks," in *Proceedings of IEEE Global Communications Conference*, 2019, pp. 1-6.
27. Y. Harikrishnan and J. He, "Clustering algorithm based on minimal path loss ratio for vehicular communication," in *Proceedings of IEE International Conference on Computing, Networking and Communications*, 2013, pp. 745-749.
28. N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Communications Surveys & Tutorials*, Vol. 21, 2019, pp. 3133-3174.
29. R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT Press, 2018.
30. V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, Vol. 518, 2015, pp. 529-533.
31. P. Gheysari, M. Fateh, and M. Rezvani, "Lung ct image segmentation using reinforcement learning," *International Journal on Artificial Intelligence Tools*, Vol. 30, 2021, p. 2150005.
32. A. Azzouni, R. Boutaba, and G. Pujolle, "Neuroute: Predictive dynamic routing for software-defined networks," in *Proceedings of IEEE 13th International Conference on Network and Service Management*, 2017, pp. 1-6.
33. Z. Zhuang, J. Wang, Q. Qi, H. Sun, and J. Liao, "Toward greater intelligence in route planning: A graph-aware deep learning approach," *IEEE Systems Journal*, Vol. 14, 2019, pp. 1658-1669.
34. J. Qadir, "Artificial intelligence based cognitive routing for cognitive radio networks," *Artificial Intelligence Review*, Vol. 45, 2016, pp. 25-96.
35. T.-V. Nguyen, T.-N. Tran, T. Huynh-The, and B. An, "An efficient qos routing protocol in cognitive radio manets: Cross-layer design meets deep reinforcement learning," in *Proceedings of IEEE International Conference on Electronics, Information, and Communication*, 2021, pp. 1-4.
36. V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of International Conference on Machine Learning*, 2016, pp. 1928-1937.
37. J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv Preprint*, 2017, arXiv:1707.06347.

38. G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv Preprint*, 2016, arXiv:1606.01540.
39. P. Hernandez-Leal, B. Kartal, and M. E. Taylor, "A survey and critique of multiagent deep reinforcement learning," *Autonomous Agents and Multi-Agent Systems*, Vol. 33, 2019, pp. 750-797.
40. A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," Technical Report, Los Alamos National Lab, Los Alamos, NM (United States), 2008.
41. G. Jayakumar and G. Gopinath, "Performance comparison of manet protocols based on manhattan grid mobility model," *Journal of Mobile Communication*, Vol. 2, 2008, pp. 18-26.
42. P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *Proceedings of IEEE 21st International Conference on Intelligent Transportation Systems*, 2018, pp. 2575-2582.
43. B. E. Bilgin and V. C. Gungor, "Performance comparison of IEEE 802.11 p and IEEE 802.11 b for vehicle-to-vehicle communications in highway, rural, and urban areas," *International Journal of Vehicular Technology*, Vol. 2013, 2013.

Lincoln Herbert Teixeira received his MS degree in Electrical Engineering and Industrial Informatics from Federal University of Technology, Paraná UTFPR, Brazil, in 2009. Currently, he is a Ph.D. student in the Faculty of Electrical Engineering and Informatics at the Budapest University of Technology and Economics, BME in Hungary.

Árpád Huszák received his MS degree in 2003 as an Electrical Engineer from the Budapest University of Technology and Economics (BME) at the Department of Telecommunications. He received Ph.D. degree in electrical and computer engineering in 2010. Currently, he is with the Department of Networked Systems and Services as an Assistant Professor. He is a member of the IEEE and HTE. His research interests focus on network protocols, mobile computing, and adaptive multimedia communication over wireless networks.