Distributed Deep Reinforcement Learning Computations for Routing in a Software-Defined Mobile Ad Hoc Network

Omar S.Almolaa[1], Prof. Dr. Manar Y.Kashmola[2]
[1]College of Basic Education, University of Telafer, Iraq and Phd student in Computer Science Dept. College of Computer science and Mathematics ,University of Mosul, Iraq,
[2]Computer Science Dept. College of Computer science and Mathematics University of Mosul,Iraq

Article History: Received: 10 November 2020; Revised 12 January 2021 Accepted: 27 January 2021; Published online: 5 April 2021

ABSTRACT

The need for reliable and flexible wireless networks has significantly increased in recent years, according to the growing reliance of an enormous number of devices on these networks to establish communications and access service. Mobile Ad-hoc Networks (MANETs) allow the wireless network to establish communications without the need for infrastructure by allowing the nodes to deliver each other's packets to their destination. Such networks increased flexibility but require more-complex routing methods. In this study, we proposed a new routing method, based on Deep Reinforcement Learning (DRL), that distributes the computations in a Software Defined Network (SDN) controller and the nodes, so that, no redundant computations are executed in the nodes to save the limited resources available on these nodes. The proposed method has been able to significantly increase the lifetime of the network, while maintaining a high Packet Delivery Rate (PDR) and throughput. The results also show that the End-to-End delay of the proposed method is slightly larger than existing routing methods, according to the need for longer alternative routes to balance the loading among the nodes of the MANET.

Keywords:

MANET, Ad Hoc Routing, SDN, Deep Reinforcement Learning, Distributed Computing

1. INTRODUCTION

With the reliability or an enormous number of users on wireless networks to establish different types of communications, including the access of digital services being provided over the internet, and the high availability of hand-held devices that are equipped with the required hardware to establish such networks, this type of communications is attracting significant attention in recent years [1]. Establishing these networks when infrastructure is available and well-defined is an easy task but significantly limits the flexibility of the network, especially the covered region, which is defined by the implemented infrastructure. To overcome such a limitation, different types of Ad-hoc networks, such as the Device to Device (D2D) [2], Internet of Things (IoT) [3], Wireless Sensor Networks (WSN), Vehicular Ad-hoc Networks (VANETs) [4] and Mobile Ad-hoc Networks (MANETs) [5], are being used to establish communications without the need for a predefined infrastructure. In these networks, each host

is required to deliver the packets that are initiated by other hosts to their destination, i.e. route these packets [6].

With the ability of the hosts, i.e. nodes, in ad-hoc networks to route their packets until they reach their destinations, the need for infrastructure is eliminated, so that, the flexibility of the networks increases significantly. However, with additional features come additional challenges, which in this case the need for complex routing techniques that can handle the making of complex decisions that take into consideration the movement of the hosts in the network [7]. Hence, several types of routing protocols are proposed for this type of wireless network, which can be categorized into three main categories, reactive, proactive and hybrid methods. In reactive methods, the source node initiates a route discovery operation in order to recognize the optimum route that the packet must follow to reach the required destination. In contrast, the hosts in the network maintain a table of the routes they can use to reach each host in the network, i.e. a routing table, so that, when a packet is initiated, the next hop is selected based on the information stored in the routing table. Different techniques are being used to recognize the optimal routes and maintain the routing tables, such as the Ad-hoc On-Demand Distance Vector (AODV) [8] and Dynamic Source Routing (DSR) [9] protocols. According to these specifications, the hosts in a network that use proactive routing methods are not required to initiate route discovery operation but are required to exchange a relatively higher amount of data in order to maintain the routing tables up-to-date. However, this feature allows hosts that use proactive routing protocols to communicate faster, as the packets are transmitted immediately, without the need for route discovery. By combining the features of the reactive and proactive methods, hybrid methods attempt to improve the performance of the network by reducing the time required to distinguish the route for a packet and the traffic overhead that is required to exchange information about the hosts in the networks. For instance, the Secure Link State Protocol (SLSP) [10] and the Zone Routing Protocol (ZRP) [11] divide the networks into clusters, where hosts information is only exchanged amongst the hosts in the same cluster while cluster information is exchanged amongst the clusters.

To further improve the flexibility of the network, Software-Defined Networks (SDNs) are being widely used in recent years, in which a controller is designated for the task of routing in the networks [12]. Unlike the Base Station (BS) in wireless networks that have a predefined infrastructure, the SDN controller does not receive the packets being sent from the host to the destination. Alternatively, it only receives the information that is required to recognize the routes that can connect the hosts to each other, so that, it defines a route when requested by the source host. The use of such a controller allows the employment of more complex techniques, as the use of such techniques without an SDN controller can exhaust the resources available on the hosts. However, the use of such a topology also increases the overhead imposed by the routing information that is being exchanged between the hosts from one side and the network controller from another [12, 13].

One of the methods that have shown significant improvement in MANETs when used for routing is Deep Reinforcement Learning (DRL) [14]. Based on a reward received from the environment after an action is executed, a neural network is trained to predict the outcome of each of these actions, so that, the actions that maximize the expected rewards are executed [15]. Several methods of DRL exist, such as Deep Q-Learning (DQN) Dueling Deep Q-Learning (DDQN) and Policy Gradients (PG) [16]. Unlike DQN and DDQN, PG predicts the probability of selecting an action, to maximize the reward, instead of the reward expected from executing that action. Hence, the DRL agent learns to approximate the policy of the environment, so that, it can provide better interaction [17]. PG has shown significantly better performance interacting with environments that require complex sequential actions before recognizing the reward

value, which makes it more suitable for the routing process, as the packet is required to pass through a set of hosts before it can be delivered to its destination. In general, DRL is being widely used for routing in MANETs but the complex computations that are required by the neural networks are expected to consume significant resources from the limited ones available on the hosts of the MANET.

In this study, we propose a new routing method for MANETs based on DRL and using SDN. The proposed method distributes the computations between the SDN and the hosts, so that, the amount of information being exchanged among the hosts and the SDN controller is minimized, as well as the computations required to be executed at the hosts. Accordingly, the proposed method provides more efficient employment of the resources available on the MANET, including the bandwidth and energy. The remainder of the paper is organized as follows: Works in the literature related to the topic being investigated in this paper are reviewed in Section 2 to illustrate the significance of the proposed method; Section 3 describes the method proposed in this paper, how MANET data is collected and presented to the DRL model; The method is evaluated in Section 4 and compared to existing routing methods that are widely used for routing in MANETs; Section 5 summarizes the conclusions of this study.

2. RELATED WORKS

With the absence of a predefined infrastructure, ad hoc networks rely on the nodes in the network to deliver the packets to their destination. Accordingly, traditional routing techniques for these networks rely on broadcasting information of each node in the network to the remaining nodes or discover the route that can be used to deliver the packet when a transmission is required. In MANETs, the periodic change of the position of the node has a significant influence on the performance of such protocols [17]. In reactive methods, in which the source node discovers the route to the destination when a packet is initiated, i.e. on-demand, a node that can be a part of the optimal route discovered to the destination node can leave the range of the adjacent nodes, according to their movements, as the payload packet arrives after route discovery. In proactive routing methods, the network can be easily flooded with the packets that hold the routing information, as this information is broadcasted by each node based on its movement. Hence, with a high number of mobile nodes high number of update packets are broadcasted [18, 19].

In addition to the need to address the previous concerns imposed by the movement on the nodes in MANETs, it is also important to balance this task among the nodes by using alternative routes that avoid the exhaustion of certain nodes. Despite the significant extension in the lifetime of the nodes in the network, balancing the load among the nodes requires knowledge about the packets that are currently being routed, which increases the complexity of the routing task and the amount of information each node is required to maintain. However, by using SDN topology, such a routing can be achieved using different balancing methods, as all the information is being stored locally in the SDN controller and is not broadcasted throughout the network similar to the user of methods such as Least Common Multiple-based Routing (LCMR) [20] and Proactive Source Routing (PSR) [21] methods. In addition to the avoidance of exhausting a node in the MANET, the method proposed in [22] has also shown the ability to avoid any congestions, as alternative routes are being used when a route is found to be busy. By evaluating different cross-layer congestion control methods, this study shows the ability of SDN networks to address the load balancing in wireless networks.

In addition to the need for load balancing, MANETs add another challenge to the routing procedure, which is imposed by the movements of the nodes in the network. A node that is

selected for a path to deliver a packet may not be within the required range when the packet arrives. Thus, it is important to take the speed and direction of movement into consideration when making such a decision, so that, a packet may be sent to a node that is expected to be within the required position upon the arrival of the packet and avoid that may not be predicted to be there. To address this challenge, several methods are used to handle the relatively complex inputs and make the required routing decisions, such as Ant-based Energy-Aware Disjoint Routing Algorithm (AEADMRA) [23], Semi-Markov Smooth and Complexity Restricted mobility model (SMS-CR) [24] and Deep Q-Learning (DQN) [14]. Despite the improved performance of the MANET when used DQN, the authors indicate that the complex computations that are required by the neural network in the DRL agent can be very exhaustive to the nodes. Additionally, as each node is required to process the information of all the nodes in the MANET, there is a huge amount of information being broadcasted in the network and most of the computations are being executed redundantly in the nodes, as each node is processing the same information using the same DRL agent. Additionally, the use of DQN is less efficient than using PG, as the DQN is trained to predict the lifetime of the network, which can dramatically delay the training process, compared to PG, in which the agent learns the policy required to deliver the packets and avoiding the exhaustion of nodes.

Artificial Intelligence (AI) has been attracting significant attention in recent years, according to its ability to automatically interact with complex environments and make the appropriate decisions. Reinforcement Learning (RL) is one of the AI fields that allow computers to interact with an environment based on the feedback from the environment, i.e. rewards, that are associated to the actions executed by the RL agent, based on its state in the environment, as shown in Figure 1. In a simple environment, the reward value for each possible action per each possible state can be measured and used to select the appropriate action, which is the action that maximizes the reward, at each state. However, in a complex environment, with an infinite number of possible states, such computations, i.e. brute force, are impossible to execute. Hence, Deep Reinforcement Learning (DRL) has emerged as a solution to this problem by using a deep neural network that approximates the required functions, i.e. predict the outcome of a state based on similar states. This approximation has enabled the use of DRL to solve several problems by using different approaches. Deep Q-Learning (DQN) is one of the DRL methods in which the reward expected for each action is predicted directly by the neural network. Despite the good performance of DQN in several applications, this method has shown limited ability to interact with environments that require a series of actions before a reward is assigned to the agent. Alternatively, Policy Gradient (PG) method predicts the probability of actions to be executed in the environment, based on the state of the agent, in order to maximize the reward value. PG has shown significantly better performance in environments that require a complex series of actions, such as [17, 25].

3. METHODOLOGY

As illustrated in the previous section, there are three main components for PG in DRL, which are the state, agent and reward. The state is collected from the environment, by the agent, in order to select the appropriate action, which is then executed in the environment. This action changes the state of the agent in the environment, which requires the environment to return a new state, as well as a reward value that describes the quality of the selected action. However, in some complex environments, the quality of these actions cannot be evaluated until a certain point is reached, i.e. a series of actions is executed before a reward value is calculated. This reward is then distributed on the executed actions based on their position in the series by using a discount factor, denoted by γ. This parameter adjusts the influence of the future reward on

the one calculated for the selected action, so that, the reward value at time instance *t* is equal to:

$$R_t = \gamma R_{t+1} \tag{1}$$

3.1. Rewards Assignement

During the training phase of the proposed method, a set of networks is simulated and used to calculate reward values for the agent, in order to improve the overall performance of the network. Initially, each of these networks is routed using AODV and OLSR protocols and the longest lifetime amongst these two protocols is selected as the control value of the MANET's lifetime. This value is used to calculate the relative lifetime of the network using the routing decisions made by the proposed method L_p, which is then used to calculate the reward value R, as shown in Eq. 2.

$$R = \frac{L_p \times 100}{\max(L_{AODV}, L_{OLSR})} \tag{2}$$

According to Eq. 2, the final reward for the agent when it achieves a lifetime similar to the longest lifetime achieved by the AODV or the OLSR, the final reward value is 100. Although this lifetime is a result of all the routing decisions made using the proposed method, the latest decisions have more influence on the lifetime of the network, as the energy remaining of the nodes becomes lower, compared to when the network is initialized. Thus, a discount factor β, separate from the DRL's γ, is assigned to the reward, so that, more emphasis is applied to the latest decisions and discounted rewards are provided to the initial decisions. These discounted results that are provided to the decisions made by the agent when the network is initialized, i.e. when it has more energy, forces the agent to seek for shortest paths to maximize the rewards, whereas the exhaustion of the nodes becomes more important as these resources become more limited by the end of the network's lifetime. Thus, the discounted reward value for the successful n^{th} set of routing decisions, which consists of multiple hops, among a total of *T* routing operations is calculated as:

$$R_t = R_n + \beta^{T-n} \times R, \tag{3}$$

where R_n is the instantaneous reward assigned to the routing operation of the n^{th} packet.
This reward is then discounted using the standard RL approach and the γ parameter for each hop selected by the agent until it reaches its destination. The instantaneous reward of a packet that is delivered to its destination is zero, so that, the reward assigned to that series of hops is equal to the discounted lifetime reward. If the packet is not delivered, a value of -100 is assigned to the instantaneous reward. Hence, the reward value assigned for each hop is calculated as:

$$R_h = R_{h,n} + \gamma^{H_t - h} \times R_t, \tag{4}$$

where H_t is the total number of hops that the packet *t* has traveled to reach its destination, *h* is the position of the hop in the path and $R_{h,n}$ is the instantaneous reward assigned to that action, i.e. hop. By default, this reward is zero, unless one of the conditions shown in Table 1 are met, the reward per each condition is assigned for $R_{h,n}$.

Table 1. Instantaneous reward values for hop selection actions.

Condition	**Reward**	**Explanation**
The packet passes through a node that it has already been through.	-10	To avoid infinite loops but still allow the packet to go back when it reaches a dead end.

The proposed method selects a node that is out of the reach of the current node for next hop.	-100	To definitely avoid such behavior in future decisions.
The selected node does not have sufficient energy to receive the packet.	-100	

3.2. State Representation and Decision Making

The state of the agent in the MANET contains two types of information, one of them is shared among the network whereas the other is specific for the next-hop selection task. Accordingly, the information regarding the overall network is sent to the SDN controller, in which feature extraction is executed. The extracted features are then broadcasted through the entire network, so that, the computations that are executed to extract these features are never repeated. Then, the node appends the information that is required to choose the next hop of the packet and selects the required node. Each node reports its position and remaining energy to the SDN whenever it travels a predefined threshold distance from its last reported position or after a predefined period of time. Periodic reporting ensures that the node reports its remaining energy even if it is not moving or moving slowly. In addition to this information, the SDN also needs specific information about the MANET, such as its dimensions and the range of the nodes in it. Finally, to select the next hop of a packet, it is required to specify the source, destination and nodes that the packet has traveled through so far.

During the training, all this information is provided to a single neural network, shown in Figure 1, for each hop selection task. Then, when the training is finished, the parts that are marked in red are populated in the nodes of the MANET, whereas the green parts are maintained in the SDN. First, the information retrieved from the nodes is distributed in a three-dimensional array and inputted to the first part of the neural network, which has U architecture similar to the U-net (UNET123). This architecture allows accurate mapping of deep features detected in the input, by passing this input through a series of convolutional layers to detect deep features and append them to the inputs of following layers. Then, a set of convolutional layers is also used to reduce the size of the inputs, similar to the use of artificial neural networks in compression, i.e. auto encoding. The output of this layer is broadcasted to the nodes of the MANET, so that, these nodes only append the information of the packet being routed and make their decisions.

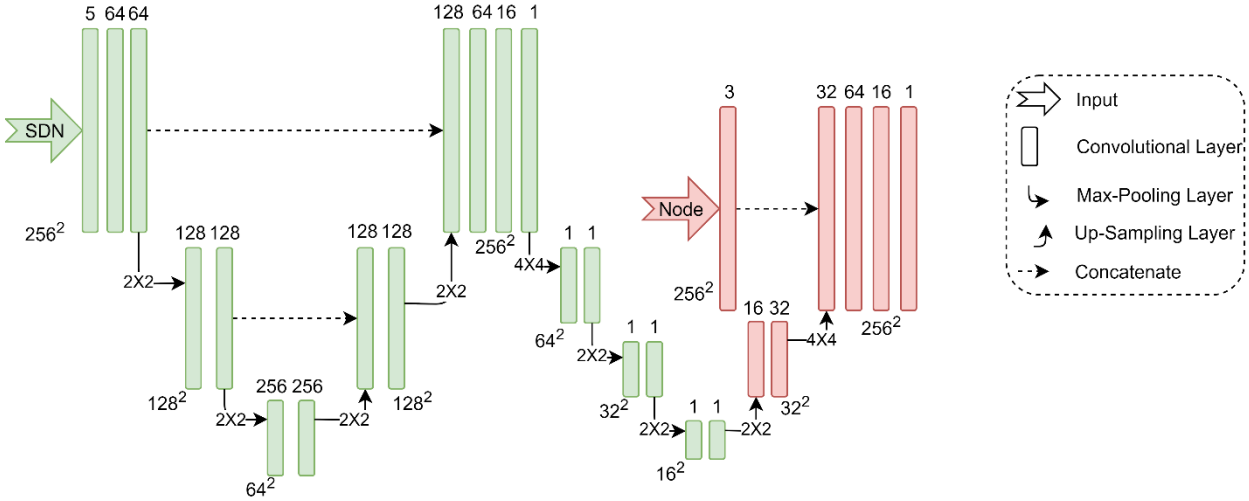


Figure 1. Illustration of the implemented neural network for the DRL agent.

To make the proposed method applicable for all MANETs, i.e. can handle any number of nodes and any dimensions, each required feature is provided using a two-dimensional array, which

produces three-dimensional input at each side, the SDN and node, as shown in Figure 3. Each input array has the dimensions of 256 × 256 values, where the feature value of each node is mapped according to the boundaries of the MANET and the position of the node. For the sample MANET shown in Figure 2, five features are prepared by the SDN, based on the information it receives from the nodes. The first array maps a value of one at each position a node with sufficient energy to receive and send a packet is located. Three other arrays present the normalized energy remaining at each node with a time span of *S* seconds, positioned according to the position received with that measurement. Accordingly, the agent can track both the energy consumption and the movement of the nodes. Finally, an array with a set of values of ones around the center of the array with a radius that represents the normalized range of the nodes.

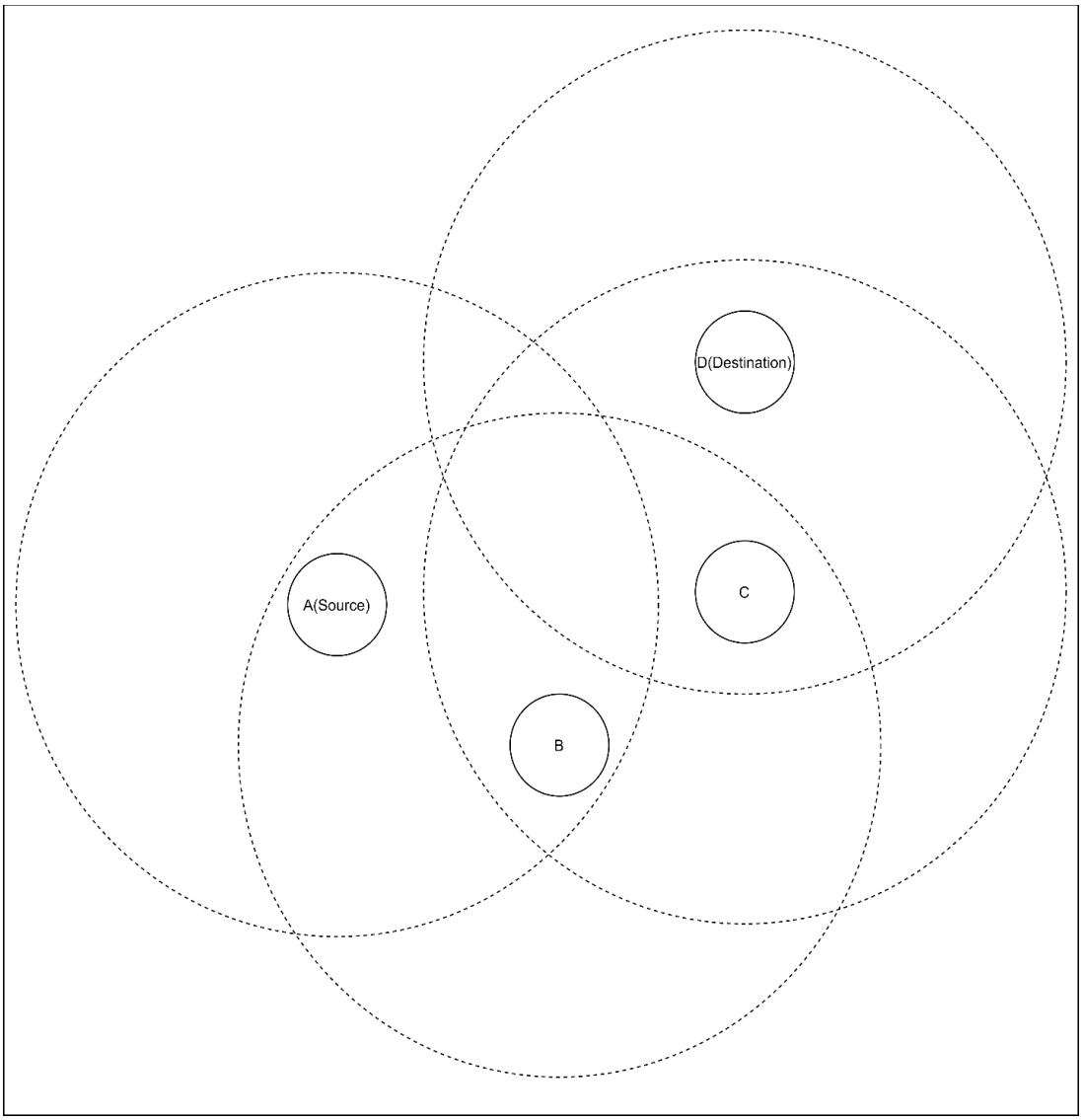
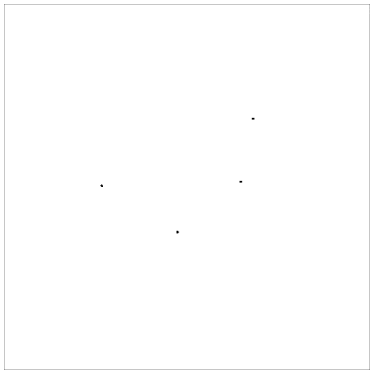
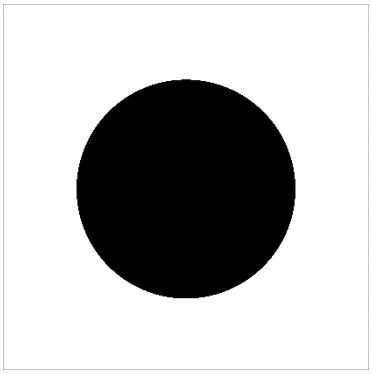


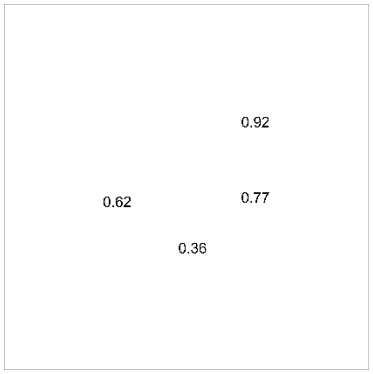
Figure 2. A sample MANET.



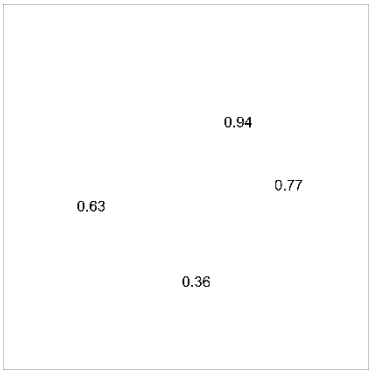
Nodes Positioning



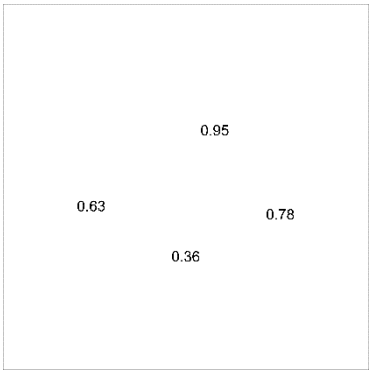
Range



Current Energy



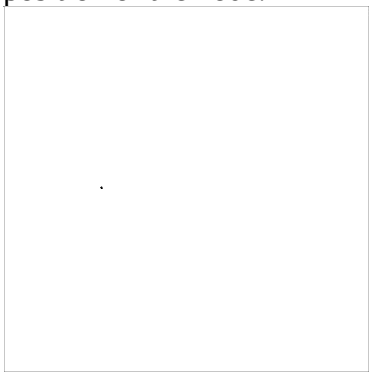
Energy at T-S



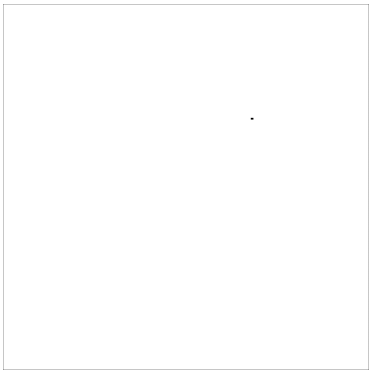
Energy at T-2S

Figure 3. SDN input to represent the sample MANET.

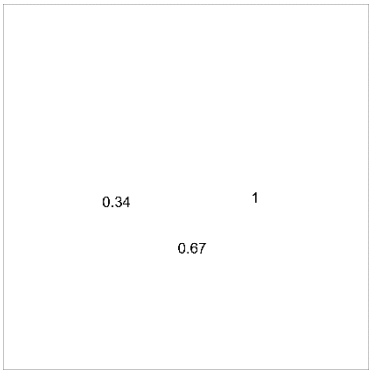
For the same sample MANET, shown in Figure 2, the node C that is currently holding the packet being sent from A to D prepares three two-dimensional arrays to represent the source, destination and the path that the packet has traveled through so far. As the node C is out of the range of the source node A, the packet must have been through node B. Hence, the inputs that node C provides to the neural network are as shown in Figure 4. The positions of the source and destination nodes are represented by positioning a value of one at the corresponding array. For the path representation, a value equal to the position of the node in the path divided by the number of hops the packet has traveled so far is positioned at the current position of the node.



Source Node Position



Destination Node Position



Packet Traveled Path

Figure 4. Node input to represent the routing task.

The output of the neural network is a two-dimensional array that represents the probability of choosing any of the nodes that are in the MANET as the next hop to deliver the packet. The node that is closest to the highest probability is selected as the next hop for the packet. Hence, this layer uses the soft-max activation function, as the summation of probabilities is always one..

3.2. Training Procedure

Despite the distribution of the computations between the SDN and the nodes, it is important to train the entire neural network of the agent at once. To provide the agent with the ability to handle different conditions, it is important to use several MANETs in the training. However, it is also important to allow the neural network to optimize its parameters, i.e. weights and biases, and evaluate the new decisions using the same MANET, as changing the MANET may naturally change the lifetime, which can confuse the neural network during training. Accordingly, to train the neural network in the proposed method, a set of 1000 randomly generated MANETs is created and saved. This procedure allows the agent to recognize different scenarios, using the different MANETs, while maintaining accurate reward values, as the same networks are being used in evaluating the actions of the agent.
In addition to these definitions, the lifetime of the MANET is discounted and used to produce reward values for each packet routing task, to emphasize the latest decisions that caused a node to exhaust its energy. Hence, the reward value calculated for a set of actions selected to route a packet at the beginning of the life of the MANET can be significantly affected by those executed by the end of its lifetime. Thus, to avoid severe effects of mistakes that can happen during the routing of the packets, a low learning rate (α) must be set for the backpropagation procedure, which is used to update the weights and biases of the neural network. Hence, a successful set of actions that efficiently deliver a packet to its destination is less affected by mistakes that may happen in future routing.

4. RESULTS AND DISCUSSION

The MANETs that are required for the training and evaluation of the proposed method as well as the neural networks of the DRL agent are implemented in Python programming language [26], using Sim2Net [27] library for MANET simulation and Tensorflow [28] library for neural networks implementation. As mentioned earlier, the proposed method is trained using 1000 randomly generated MANETs, with the characteristics shown in Table 2. The agent is trained for 1000 iterations using the entire set of training MANETs with a learning rate α of 0.0001 and discount factors γ and β of 0.99. The performance of the proposed method is illustrated by comparing the average lifetime, throughput, End-to-End (E2E) Delay and Packet Delivery Rate (PDR) for the evaluation MANETs when using the proposed method to the use of the AODV and OLSR protocols at different nodes speeds.

Table 2. Simulation parameters.

Parameter	Value
Medium Access Control (MAC) layer protocol	IEEE 802.11
Physical layer model	PHY 802.11b
Channel frequency	2.4 GHz
Transmission range	270 m
Battery capacity	3600 mAh
Battery model	Linear
Signal transmission power	31.623 mW

Generic energy model	$P_{Receive}$: 900 mW; $P_{Transmission}$: 1300mW
Traffic type	FTP(TCP)
Node movement	Random waypoint
Random waypoint parameters	Minimum velocity: 10 m/s; Maximum velocity: 60m/s.
Pause time	10 s
MANET region	Square
Region dimension	Min: 500 m; Max: 1500 m.
Number of nodes	Min: 10; Max: 100.
Packet size	512 bytes.

4.1. Network Lifetime

The average lifetime of the 100 simulated MANETs is calculated for each simulated speed using the three protocols selected for the evaluation. The results illustrated in Fig. 5 show that the proposed method has achieved a significantly higher lifetime, compared to the use of AODV and OLSR protocols. This improvement is according to the lower overhead required by the packets that deliver the routing information, to the SDN, and the network state descriptor, broadcasted from the SDN to the nodes. Additionally, the proposed method has also shown less influence by the velocity of the nodes, i.e. the lifetime does not dramatically drop when the velocity is increased, as in the use of the other protocols. This behavior is according to the ability of the proposed method to consider the movement of the nodes in the decision-making.

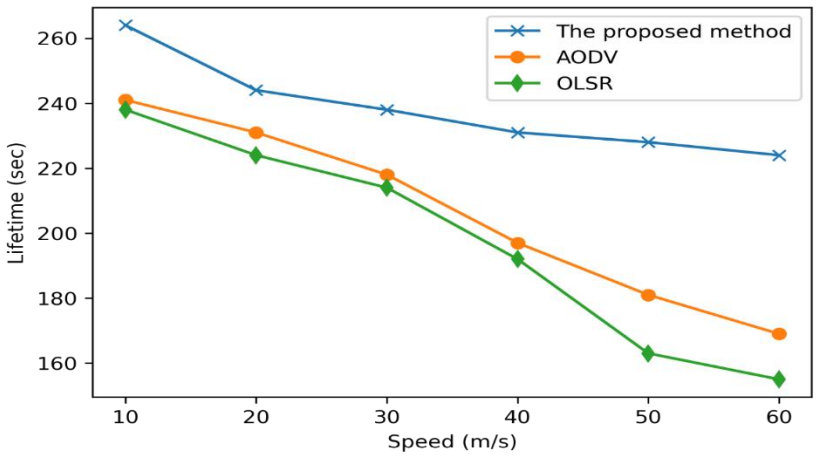


Figure 5. Average MANETs lifetime versus speed of nodes.

4.2. Network Throughput

The throughput of a network is equal to the number of payload bits delivered in a second, which can be calculated using the formula shown in Eq. 5. This throughput is measured for the evaluation MANETs at different nodes speeds, as summarized in Fig. 6. The results show that the MANET has achieved higher throughput when using the proposed method for routing. This behavior is according to the less overhead required by the proposed method, which allows more of the MANET's bandwidth to be used to communicate the payload packets, as well as the effectiveness of the method is delivering the packets. Additionally, the AODV protocol has also achieved higher throughput, compared to the OLSR protocol, similar to the results illustrated in [29]. Additionally, Fig. 6 also shows that the proposed method has shown less influence by the speed of the nodes, compared to the use of the standard protocols. This ability is a result of the use of historical data that allows the agent to recognize the speed and path that each node is traveling. Hence, a node can be selected based on its predicted position when the packet arrives rather than its current position, which is the only position that is taken into consideration in the existing protocols.

$$Throughput = \frac{Number\ of\ delivered\ packets \times 512 \times 8}{Operation\ time\ in\ seconds} \quad (5)$$

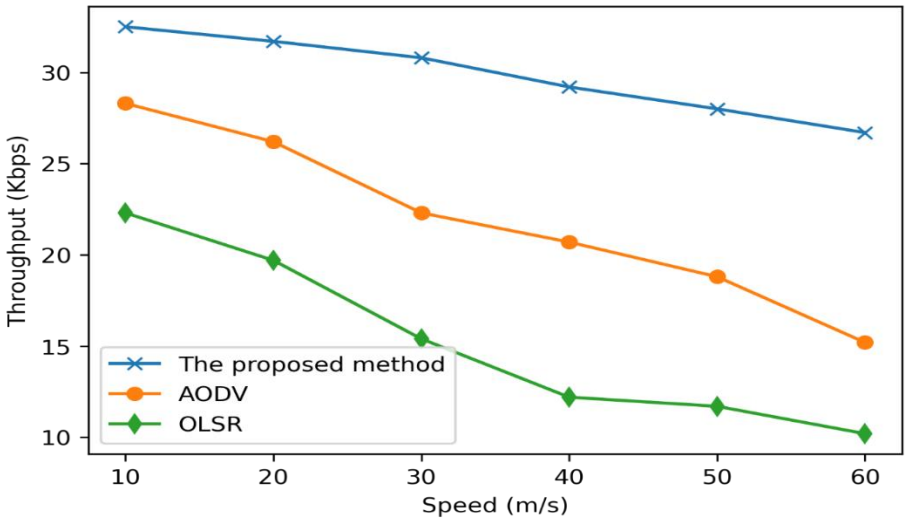


Figure 6. Average MANETs throughput versus node velocity.

4.3. End-to-End Delay

The E2E delay is the time required by a packet to reach its destination. The average E2E delays for the evaluation MANETs is computed and illustrated in Fig. 7. The results show that the proposed routing protocol has not been able to reduce the E2E delay, according to the need to use alternative, longer, routes to avoid the use of exhausted nodes. However, unlike the results reported in [29], the results show that the OLSR protocol has achieved faster delivery, i.e. shorter E2E delay, which is an expected behavior according to the need for route discovery when transmitting packets in MANETS that use AODV routing protocol, similar to any reactive routing protocol. Moreover, these results also defy the hypothesis in [14], as the more power-efficient protocol is obliged to use longer alternative routes to balance the loading amongst the nodes. However, their results show that the proposed method based on Q-Learning has achieved lower E2E delay. Finally, the results also shows that reactive protocols can achieve better routes at faster nodes, as these nodes may change their positions way faster than the updates required in the proactive methods.

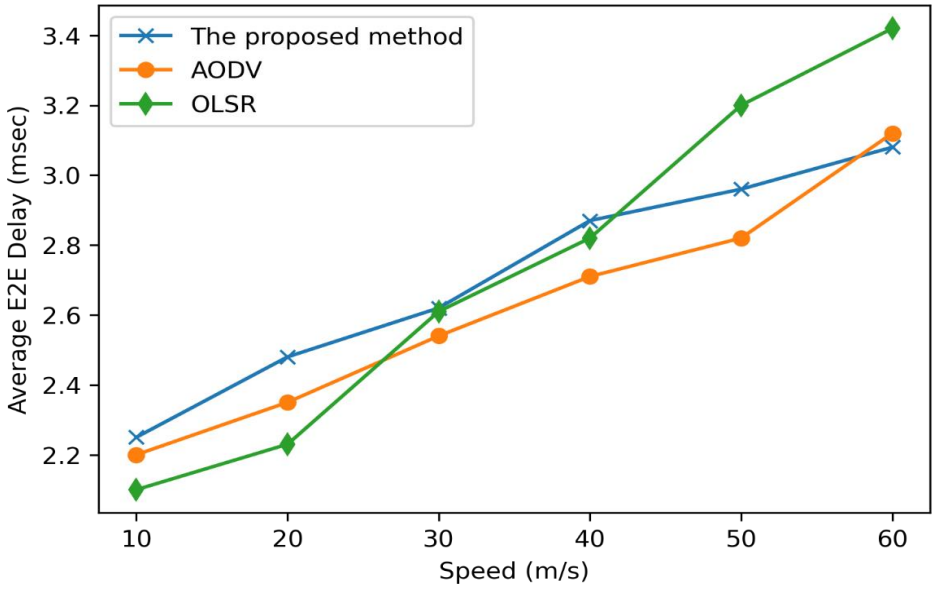


Figure 7: Average E2E Delay for the simulated MANETs.

4.4. Average Packet Delivery Rate (PDR)

The average PDR is calculated for the proposed method and summarized in Fig. 8, which shows that the proposed method has achieved high PDR, compared to the existing methods. Fig. 8 also shows that the proposed method has shown almost no influence by the node speed to the PDR, according to the ability of the proposed method to recognize the direction and velocity of each node. Additionally, the larger gap between the proposed method and the use of Q-Learning in [14] shows that the PG is more suitable for the required task, as it has been able to deliver significantly more packets, which is the main aim of a routing protocol.

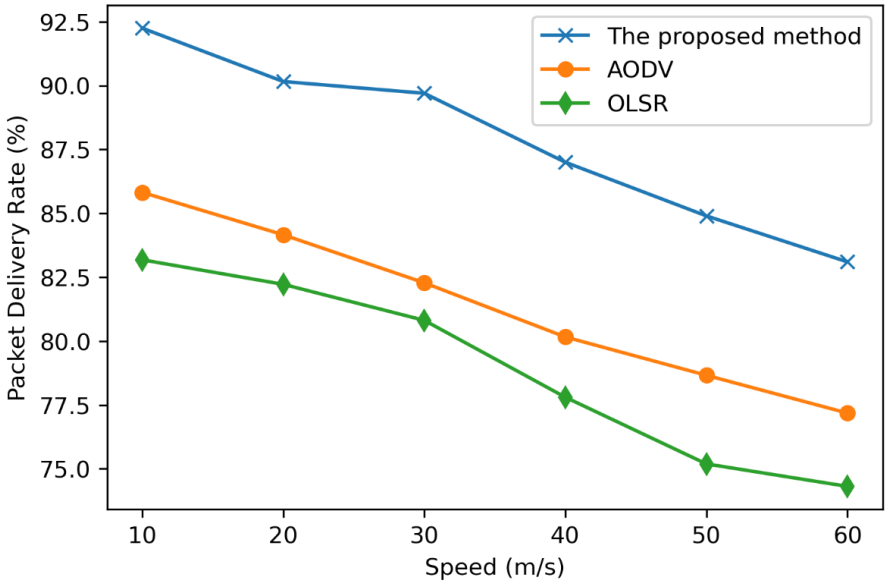


Figure 8. Packet delivery rate for the simulated MANETs versus nodes speed.

Provide a statement that what is expected, as stated in the "Introduction" chapter can ultimately result in "Results and Discussion" chapter, so there is compatibility. Moreover, it can also be added the prospect of the development of research results and application prospects of further studies into the next (based on result and discussion).

5. CONCLUSION

With the rapidly growing reliance on wireless communications to establish different types of connections and access a variety of services, the predefined infrastructure of these networks has become the limit to their operation. Ad-hoc networks have emerged as a solution for this problem by allowing the nodes in the network to establish communications by delivering each other's packets. However, the absence of infrastructure and the ability of the nodes in the network to move has brought significant challenges toward routing the packets in the network. One of the main concerns in these networks is the efficient use of the limited resources on the nodes, in order to extend the lifetime of the network.

In this study, we propose a new routing method for MANETs based on reinforcement learning. The proposed method aims to balance the loading among the nodes, so that, nodes with limited energy remaining in their power sources are avoided, even if the packet is required to travel a longer path. However, according to the complex computations required by reinforcement learning and the need for network information to be processed every time a packet is being routed, the computations are distributed between the SDN controller and the nodes, so that, the nodes send their information to the SDN, which calculates a vector that describes the network

and broadcast it to the nodes. The nodes then complete the computations that are specifically required for the packet and select the next hop. The proposed method has shown significant extension in the lifetime of the network, while maintaining higher throughput and PDR, compared to the existing protocols. However, the results show that the E2E delay of the proposed method has been similar, which is according to the need for using longer alternative routes to avoid exhausting the nodes.

In future work, the ability to use Deep Deterministic Policy Gradient (DDPG), which is another type of reinforcement learning, is going to be investigated. With its ability to output linear values that maximize the reward value, the use of DDPG can further reduce the computations executed in the nodes by avoiding the use of convolutional layers and directly output the mapped position of the candidate next hop, using only dense layer. Dense layers are significantly less-complex than convolutional ones, which can further improve the lifetime of the MANET.

REFERENCES

[1] V. Yazıcı, U. C. Kozat, and M. O. Sunay, "A new control plane for 5G network architecture with a case study on unified handoff, mobility, and routing management," *IEEE Communications Magazine,* vol. 52, pp. 76-85, 2014.

[2] M. N. Tehrani, M. Uysal, and H. Yanikomeroglu, "Device-to-device communication in 5G cellular networks: challenges, solutions, and future directions," *IEEE Communications Magazine,* vol. 52, pp. 86-92, 2014.

[3] O. Elijah, T. A. Rahman, I. Orikumhi, C. Y. Leow, and M. N. Hindia, "An overview of Internet of Things (IoT) and data analytics in agriculture: Benefits and challenges," *IEEE Internet of Things Journal,* vol. 5, pp. 3758-3773, 2018.

[4] C.-F. Huang, Y.-F. Chan, and R.-H. Hwang, "A comprehensive real-time traffic map for geographic routing in vanets," *Applied Sciences,* vol. 7, p. 129, 2017.

[5] X. Wang and J. Li, "Improving the network lifetime of MANETs through cooperative MAC protocol design," *IEEE transactions on parallel and distributed systems,* vol. 26, pp. 1010-1020, 2013.

[6] A. Chopra and R. Kumar, "Efficient Resource Management for Multicast Ad Hoc Networks: Survey," *International Journal of Computer Network and Information Security,* vol. 8, p. 48, 2016.

[7] S. Boussoufa-Lahlah, F. Semchedine, and L. Bouallouche-Medjkoune, "Geographic routing protocols for Vehicular Ad hoc NETworks (VANETs): A survey," *Vehicular Communications,* vol. 11, pp. 20-31, 2018.

[8] W.-K. Kuo and S.-H. Chu, "Energy efficiency optimization for mobile ad hoc networks," *IEEE Access,* vol. 4, pp. 928-940, 2016.

[9] A. Taha, R. Alsaqour, M. Uddin, M. Abdelhaq, and T. Saba, "Energy efficient multipath routing protocol for mobile ad-hoc network using the fitness function," *IEEE access,* vol. 5, pp. 10369-10381, 2017.

[10] S. Rosati, K. Krużelecki, G. Heitz, D. Floreano, and B. Rimoldi, "Dynamic routing for flying ad hoc networks," *IEEE Transactions on Vehicular Technology,* vol. 65, pp. 1690-1700, 2015.

[11] D. Hurley-Smith, J. Wetherall, and A. Adekunle, "SUPERMAN: security using pre-existing routing for mobile ad hoc networks," *IEEE Transactions on Mobile Computing,* vol. 16, pp. 2927-2940, 2017.

[12] C. Y. Hans, G. Quer, and R. R. Rao, "Wireless SDN mobile ad hoc network: From theory to practice," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1-7.

[13] K. S. Atwal, A. Guleria, and M. Bassiouni, "SDN-based mobility management and QoS support for vehicular ad-hoc networks," in *2018 International Conference on Computing, Networking and Communications (ICNC)*, 2018, pp. 659-664.

[14] V. Tilwari, K. Dimyati, M. Hindia, A. Fattouh, and I. S. Amiri, "Mobility, residual energy, and link quality aware multipath routing in MANETs with Q-learning algorithm," *Applied Sciences,* vol. 9, p. 1582, 2019.

[15] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *arXiv preprint arXiv:1811.12560,* 2018.

[16] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," *Expert Systems with Applications,* vol. 141, p. 112963, 2020.

[17] C. Daskalakis, D. J. Foster, and N. Golowich, "Independent Policy Gradient Methods for Competitive Reinforcement Learning," *Advances in Neural Information Processing Systems*, vol. 33, 2020.
[18] Y. Bai, Y. Mai, and N. Wang, "Performance comparison and evaluation of the proactive and reactive routing protocols for MANETs," in *2017 Wireless Telecommunications Symposium (WTS)*, 2017, pp. 1-5.
[19] Y. Jahir, M. Atiquzzaman, H. Refai, A. Paranjothi, and P. G. LoPresti, "Routing protocols and architecture for disaster area network: A survey," *Ad Hoc Networks*, vol. 82, pp. 1-14, 2019.
[20] A. Bhattacharya and K. Sinha, "An efficient protocol for load-balanced multipath routing in mobile ad hoc networks," *Ad Hoc Networks*, vol. 63, pp. 104-114, 2017.
[21] Z. Wang, Y. Chen, and C. Li, "PSR: A lightweight proactive source routing protocol for mobile ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 63, pp. 859-868, 2013.
[22] Q.-V. Pham and W.-J. Hwang, "Network utility maximization-based congestion control over wireless networks: A survey and potential directives," *IEEE Communications Surveys & Tutorials*, vol. 19, pp. 1173-1200, 2016.
[23] Z.-Y. Wu and H.-T. Song, "Ant-based energy-aware disjoint multipath routing algorithm for MANETs," *The Computer Journal*, vol. 53, pp. 166-176, 2010.
[24] Z. Li and Y. Wu, "Smooth mobility and link reliability-based optimized link state routing scheme for MANETs," *IEEE Communications Letters*, vol. 21, pp. 1529-1532, 2017.
[25] Z. Zhang, Y.-S. Ong, D. Wang, and B. Xue, "A Collaborative Multiagent Reinforcement Learning Method Based on Policy Gradient Potential," *IEEE transactions on cybernetics*, 2019.
[26] M. F. Sanner, "Python: a programming language for software integration and development," *J Mol Graph Model*, vol. 17, pp. 57-61, 1999.
[27] M. Kalewski, "Simple Network Simulator (sim2net) Documentation," 2017.
[28] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265-283.
[29] M. A. Jubair, S. H. Khaleefah, A. Budiyono, S. A. Mostafa, and A. Mustapha, "Performance evaluation of AODV and OLSR routing protocols in MANET environment," *Int. J. Adv. Sci. Eng. Inf. Technol*, vol. 8, pp. 1277-1283, 2018.