Multi-agent Reinforcement Learning Based Approach for Vehicle Routing Problem

Jagdeep Singh[1], Sanjay Kumar Dhurandher[2], Isaac Woungang[3](✉), and Telex Magloire N. Ngatched[4]

[1] Department of Computer Science and Engineering, Sant Longowal Institute of Engineering and Technology, Longowal, India
[2] Department of Information Technology, Netaji Subhas University of Technology, New Delhi, India
[3] Department of Computer Science, Toronto Metropolitan University, Toronto, Canada
iwoungan@ryerson.ca
[4] Faculty of Engineering and Applied Science, Grenfell Campus, Memorial University, St. John's, Canada
tngatched@grenfell.mun.ca

Abstract. Multi-Vehicle routing to service consumers in dynamic and unpredictable surroundings such as congested urban areas is a difficult operation that needs robust and flexible planning. Value iteration networks hold promise for planning vehicle routing problems. Conventional approaches aren't usually constructed for real-life settings, and they are too slow to be useful in real time. In comparison, the Vehicle Routing Problem with Value Iteration Network (VRP-VIN) offers a neural network model based on graphs that can execute multi-agent routing in a highly dispersed but connected graph with constantly fluctuating traffic conditions using learned value iteration. Furthermore, the model's communication module allows vehicles to work better in a cooperative manner online and can easily adapt to changes. A virtual environment is constructed to simulate real-world mapping by self-driving vehicles with uncertain traffic circumstances and minimal edge coverage. This method beats standard solutions based on overall cost and run time. Experiments show that the model achieves a total cost difference of 3% when compared with a state-of-art solver having global information. Also, after being trained with only 2 agents on networks with 25 nodes, can easily generalize to a scenario having additional agents (or nodes).

Keywords: Reinforcement learning · Vehicle routing problem · Value iteration networks · Graph attention layer · Multi-agent communication

1 Introduction

As vehicles grow increasingly widespread, one of the most basic issues is understanding how to navigate a fleet of vehicles to perform a specified job. Also, the huge population densities in our cities today put all existing infrastructure, especially urban transportation networks, under strain. With the progression of services like e-commerce and vehicle sharing, these congested cities' transportation demands have also gotten more complicated. So, it is very important to route vehicles in way so as to reduce overall cost, time, and congestion. Different methods [1] have been proposed to route vehicles. One of the classic methods in which a single agent is entrusted with determining the shortest path between a set of sites and destinations is known as Travelling Salesperson Problem. The multi-agent approach to this problem is called the Vehicle Routing Problem (VRP) [2]. In VRP, multiple agents try to find an optimal route by visiting a set of locations exactly once. Even after having a huge number of solvers, they are primarily built to perform planning offline and cannot modify solutions when used online. They are, however, often evaluated on simple planar network benchmarks with limited exploration in multi-agent environments. Furthermore, none of these solutions were created for dynamic contexts where online communication may be quite advantageous.

The Value Iteration Networks [3] have excellent planning capability and can generalize better in a diverse set of tasks. Its purpose is to discover a policy that optimizes expected returns. The value function peak at the goal, so the high-value function mean the destination. In the attention mechanism, only a subset of the input characteristics (value function) is meaningful for a specific label prediction (action). It is also well known that attention improves learning performance by lowering the effective number of network parameters used during learning. We have given a fleet of cars in a multi-agent environment. We have to determine the minimum total cost for mapping a given graph under traffic conditions, such that all routes are traversed not less than a defined number, and this number is not known prior. The Vehicle Routing Problem value iteration network is a distributed neural network designed for managing multiple vehicles intended to complete a specific task. Each agent has a value iteration module to carry out its own planning with the help of communication between agents via an attention mechanism. The dense adjacency matrix [4] is used for encoding paired edge information to accelerate information sharing and allow for more complex encoding since our focus is on sparse road graphs. Using actual traffic flow simulation, we illustrate the usefulness of VRP-VIN on actual road maps derived from eighteen different cities around the world. A random sub-graph of those cities was used to produce training and evaluation examples comprising real-world mapping difficulties, and then a random number is selected, which determines how many times each node in each graph is covered [5,6]. The fleet will be unaware of this knowledge until they reach this number. We use the total time taken for traversal as our major evaluation criterion, demonstrating that this technique outperforms both conventional VRP solvers and recently suggested deep learning models. Moreover, VRP-VIN adapted effectively to the graph size and agent count.

The paper is organized as follows: Sect. 2 gives us detailed literature on value iteration networks. The proposed Model is presented in Sect. 3. An evaluation of the proposed model is available in Sect. 4. Finally, Sect. 5 represents the conclusion and future work of the proposed work.

2 Related Work

In [8], Tamar et al. proposed a neural network incorporated with a planning module. They can learn to plan and can anticipate planning-related outcomes, such as reinforcement learning. They are based on a differential estimation of the value-iteration algorithm using CNN. Value iteration is a technique based on the Markov decision process. The MDP's purpose is to discover a policy that in turn optimizes our expected return. V_n (state value function at iteration n) converges to V* (ideal state value function) using the value iteration technique as n approaches infinity [6]. The VI module in the VIN takes advantage of the fact that each iteration of VI can be visualized as previous V_n and the reward function passes through a convolutional and max pooling layer. The Q function for every channel in the convolution layer refers to a specific action. As a result, K iterations of VI are equivalent to K times of applying a convolutional layer.

In [9], Lu et al. suggested a distributed cooperative routing method (DCR) based on evolutionary game theory to coordinate vehicles. This solution combines edge computing and intelligence to run on roadside units. Nash equilibrium is achieved under DCR. No vehicle can find a path more suitable than the one currently under Nash equilibrium [7].

In [10], Tang et al. proposed a reinforcement learning model with multi-agents for a centralized vehicle routing in order to improve the spatial-temporal coverage. Two reinforcement learning: proximal policy optimization and deep q-learning have been used to create routing policies. A centralized routing method is proposed for vehicular mobile crowd-sensing systems (VMCS) to expand their range of sensing based on MARL. The author initiates by customizing an environment for reinforcement learning in order to get the maximum feasible spatial-temporal coverage based on user preferences for various regions. They designed two MARL algorithms based on the Deep Q network [14] and Proximal Policy Optimization (PPO) [15]. Then, they do comparisons and sensitivity analyses to figure out how well the two methods work for VMCS problems.

In [11], Niu et al. Proposed a Multi-Agent Graph-attention Communication (MAGIC). It is a novel multi-agent reinforcement learning algorithm with a graph-attention communication protocol having a Scheduler to aid the challenge of when and to whom messages should be sent, and a Message Processor employs Graph Attention Networks (GATs) [12] comprising dynamic graphs for handling communication signals. A combination of a graph attention encoder and a differentiable attention mechanism [16] is used to develop the scheduler that provides dynamic, differentiable graphs to the Message Processor, allowing the Scheduler and Message Processor to be trained at the same time.

3 System Model

In this section, the solution to the Vehicle Routing Problem with Value Iteration Network has been discussed in detail. Vehicle Routing Problem with Value Iteration Network (VRP-VIN) has two main components:

– Asynchronous communication module [17] saves messages sent by agents in a temporary unit and retrieves information via agent-level attention method. This information is received by the value iteration network for path planning in the future.
– Value iteration network operates locally on each node repeatedly to calculate the value of traveling to each node for its next route. After that, LSTM [20] planning unit with attention mechanism repeatedly refines the node features and produces a value function associated with each node. The next destination will be selected on the basis of the value function, so the node with the highest value function will become the next destination.

Figure 1 represents the flowchart of the proposed model VRP-VIN The proposed VRP-VIN model is dynamic in nature. In order to do this, VRP-VIN includes a communication module based on the attention mechanism, in which attention is now focused on the agents as opposed to the street segment earlier in the VI module. Whenever an agent acts, it outputs some communication vector: $y^{(i)}$. It is subsequently transmitted to each agent using $Z^{(K)}$, the value iteration module's final encoding. The communication vector [18] is expressed as a set of node attributes in order to obtain the topology of the street graph. At the receiver end, each sender's current communication vector is stored temporarily. Every agent has an attention layer that compiles data from the receiver's inbox, whenever an agent wants to take a new action.

The Routing Path [19] can be represented as a strongly connected graph (G) with edges (E) and vertices (V), where we want to generate a routing path for D agents $\{R^{(i)}\}_{i=1}^{D}$, and each vertex in the graph is traversed D_v time across all agents. Until a specific number is reached, D_v is unknown to the agents, and local traffic information [13] is the only thing that can be observed. Each agent collects surrounding environment observation and information gathered from other agents and then outputs the next step's route.

A route is defined as a sequence of action $R(i) = [s_0^i, \ldots.., s^N]$, where s_0^i represents routing steps taken by agent 'i' in time t, indicating the next node to traverse, and each step represents an intermediary destination [20]. The strategy of a single agent can be described as a function of the graph of the road network; surrounding environment observation b_t^i; the communication messages sent by other agents y_t^j; and, the current status of an agent j_t^i. The mathematical Eq. 1 are as follows:

$$s_t^i, y_t^j = f(G, b_t^i, \{y_{t-1}^j\}_{j=1}^{M}; j_t^i), \tag{1}$$

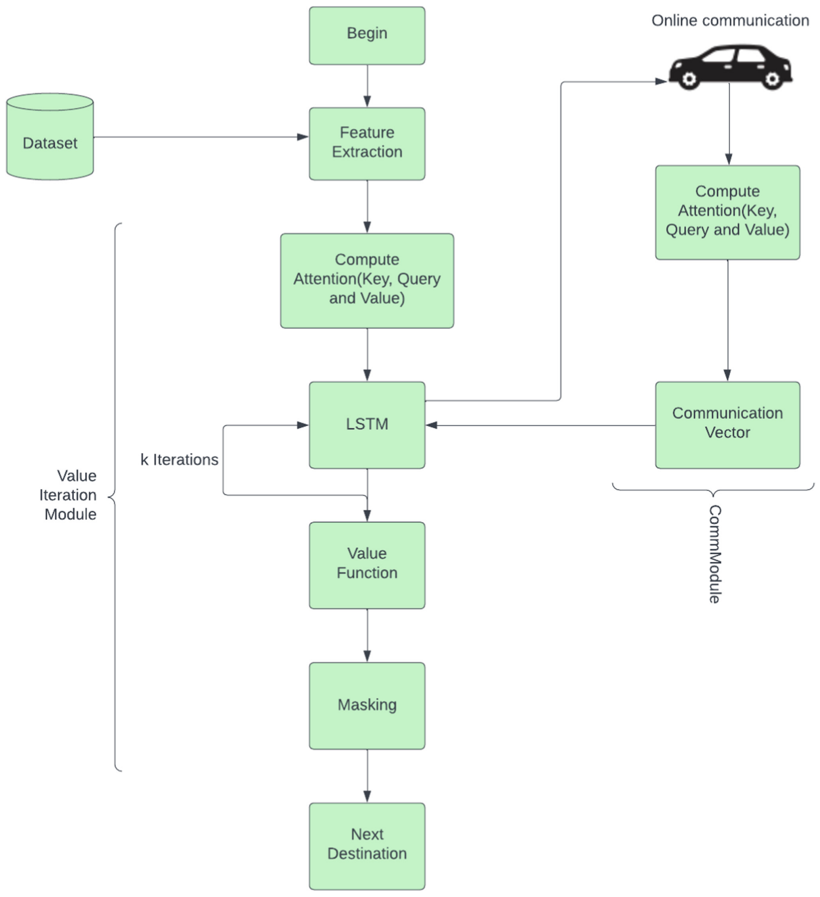

Fig. 1. Flowchart of proposed VRP-VIN Model

Consider a traffic model D determines how long it takes to travel a route, we want our system to accomplish the following goal:

$$min_R(i) \sum_{X=1...L} D(R^i), \tag{2}$$

subject to

$$\sum_i T(R^i, v) \geq T_v, v \tag{3}$$

where $T(R, v)$ tells how many times a node v should be visited in path R.

It is worth noting that the model is resilient and failure-proof as the model runs locally on all the agents, which allows it to scale better with the number of agents.

Symbol	Representation
t	Current timestamp
G	Graph of road map
L	No of Agents
n	No of graph nodes
f	Policy of Routing and Communication
π	Routing Policy
F	Time Cost given a route R
o	Agent i's observation at time t
s	Agent i state at time t
a	Agent i action at time t
m	Message vector sent by agent 'i' at time t
D_v	No of times node v needs to be visited
Z_i	At the kth value iteration, Agent i's node feature
Y_i	Agent i's input communication feature

The road network [21] is represented by a tightly linked graph G(V, E). Each graph node represents a street segment, and each agent's goal is to choose a node to be its next destination. Initial node features are refined by passing them through a graph neural network [22] for specific iterations. Then, these features are turned into a value function, and the next destination is the node with the highest value. Let $Z = (z_1, z_2, ..., z_n)$ represents a vector with initial node features, where n denotes number of nodes, and $Y = y_1, y_2, ..., y_n$ denote the node features of the input communication. A linear layer encodes node input features to produce an initial feature for the value iteration module [2]:

$$Z^0 = (Z||Y)W_{enc} + b_{enc} \tag{4}$$

We conduct the following iterative update across neighboring nodes at each planning iteration 'z' using an attention LSTM:

$$Z^{(k+1)} = Z^{(k)} + LSTM(Att(Z^{(k)}, A); H^{(k)}) \tag{5}$$

where K denotes the number of value iteration steps, hidden states $H^{(t)}$ in LSTM, and adjacency matrix A. Floyd Warshall method [23] is employed to compute dense distance matrix, which is then used as an input to this model, rather than using the adjacency matrix as an input to the network. This ensures that our model uses more useful information. The Floyd-Warshall algorithm generates a matrix, $D_{i,j} = d(v_i, v_j)$, which represents the shortest path between any two nodes in terms of pairwise distance. This matrix is then normalized to create a dense adjacency matrix. $A = (D - \mu)/\sigma$ where μ is the mean of the elements of D and σ is the standard deviation of the elements.

The graph attention layer(GAL) [24] is responsible for the exchange of information within a graph. The attention module used in VRP-VIN is a transformer layer that receives adjacency matrix and node features, then outputs modified features. First, the values of the key, query, and value function for each node are calculated.

$$Q^{(k)} = Z^{(k)}W_q + b_q, \tag{6}$$

$$K^{(k)} = Z^{(k)}W_k + b_k, \tag{7}$$

$$V^{(k)} = Z^{(k)}W_v + b_v \tag{8}$$

The node feature vector is multiplied by the weight vector to calculate the key, query, and value. Then we form an attention matrix A_{att} by computing attention between the node and each other nodes.

$$A_{att} = Q^{(k)}.K^{(k)T} \tag{9}$$

To express edge features, we mix adjacency matrix A and attention matrix A_{att} in a multi-layer neural network g as shown below.

$$A(k) = softmax(g(A_{att}^{(k)}, A)) \tag{10}$$

The values of new nodes are calculated by merging the values generated by the other nodes in the merged attention matrix according to the attention. The output of GALs is sent into the LSTM module.

$$Z^{(k+1)} = Z^{(k)} + LSTM(A^{(k)}V^{(k)}; H^{(k)}) \tag{11}$$

Before decoding, the entire procedure is performed for a fixed no. of iterations, k = 1. . . ..K.

Each node feature is translated into a scalar value function on the graph after iterating the attention LSTM module for K iterations. Then SoftMax function is applied across the rest of the nodes to derive action probabilities, masking off the value of any node that is not required to be visited anymore because they're fully traversed.

$$\pi(s_i; j_i) = softmax(Z^K W_{dec} + b_{dec}) \tag{12}$$

Now, the node with the highest probability value is chosen as the next destination. With the help of the shortest path algorithm, a full path is constructed by linking the latest node with the node chosen as the next destination. The graph weight is calculated by dividing the length of a road segment by the average speed of the car driving it. It shows how long it is expected to take to drive from one road segment to the next.

There are 22,814 directed road graphs in the collection, which were collected from 18 cities on six continents. For testing, we pick a different location, and for validation, we utilize 10% of the training set. We also include actual traffic situations and mapping issues in this benchmark. Extra problems fall into three groups: random revisits, realistic traffic, and asynchronous execution (Figs. 2, 3 and 4).

4 Evaluation

A 3-layer MLP with 16 dimensions each, with ReLU activation is utilized to combine the dot-product attention with the distance matrix. The encoding vectors have a 16-dimensional size. Adam optimizer is used to set the model's learning rate during training to be 1e−3. The decay rate is set as 0.1 per 2000 epochs. The model is trained for 5000 epochs. Each of the 50 graphs in our batch size has a maximum of 25 nodes. We just use two agents to train our network and one to nine agents to analyze it [25]. LKH3 is the best-performing iterative solver with available data. First, the solver chooses the best route for precisely covering all nodes exactly once. After that, the solver determines a new optimal route across each of the nodes that still need to be traversed. Until every node has been completely mapped, this is repeated. Basically, the solver does VRP traversals until the required number of nodes has been visited. If an agent had been given global knowledge of every hidden state, this is the best possible performance that could have been obtained. By giving the LKH3 solver information about every hidden variable and doing an optimal plan search, the solution is discovered. By duplicating the nodes and raising the node's edge weights affected by traffic congestion, we alter the adjacency matrix. GATs [12] exchange complex information between the nodes based on the attention mechanism. Normally, GAT architectures, on the other hand, presume that all edges have equal weight rather than encoding the information about the distance matrix, which restricts their potential. Although GATs aren't always made to address TSP or VRP issues, they are still among the most cutting-edge options for graph and network encoding (Tables 1 and 2).

Table 1. Total cost (runtime in hrs)

	No of vehicles			
No of nodes	1	2	3	4
20	1.1	1.1	1.4	1.9
30	1.2	1.4	1.8	2.2
40	1.4	1.6	2	2.6
50	2	2.2	2.6	3.2
60	2.1	2.3	2.7	3.3
70	2.2	2.4	2.8	3.5
80	2.6	2.9	3.2	4
90	2.9	3	3.6	4.2
100	3.5	3.8	4.3	5.1

The performance of VRP-VIN is the best over a range of agent counts and graph sizes. Notably, this technique with 25 nodes and two agents using reinforcement learning achieves a total cost of around 3% when compared with oracle.

Table 2. Average traversal cost on real graphs; Time cost (hrs); Runtime (ms)

	25 Nodes, 1 Vehicle				25 Nodes, 2 Vehicles		
Method	Cost	Gap	Runtime	Method	Cost	Gap	Runtime
Oracle	1.16	0.00%	71.3	Oracle	1.28	0.00%	438
LKH3	1.26	8.84%	71.2	LKH3	1.8	40.50%	438
GAT	1.53	32.50%	43	GAT	1.56	21.60%	29.1
VRP-VIN(IL)	1.37	18.00%	62.8	VRP-VIN(IL)	1.42	11.30%	66.6
VRP-VIN(RL)	1.25	8.17%	62.8	VRP-VIN(RL)	1.32	2.87%	56.6
	50 Nodes, 2 Vehicles				100 Nodes, 5 Vehicles		
Method	Cost	Gap	Runtime	Method	Cost	Gap	Runtime
Oracle	1.85	0.00%	902	Oracle	3.19	0.00%	2430
LKH3	2.54	37.30%	902	LKH3	6.14	92.50%	2430
GAT	2.58	39.70%	38	GAT	5.43	70.20%	38.2
VRP-VIN(IL)	2.21	19.00%	71.5	VRP-VIN(IL)	4.36	36.70%	72.8
VRP-VIN(RL)	2.12	14.50%	71.4	VRP-VIN(RL)	4.62	44.90%	72.8

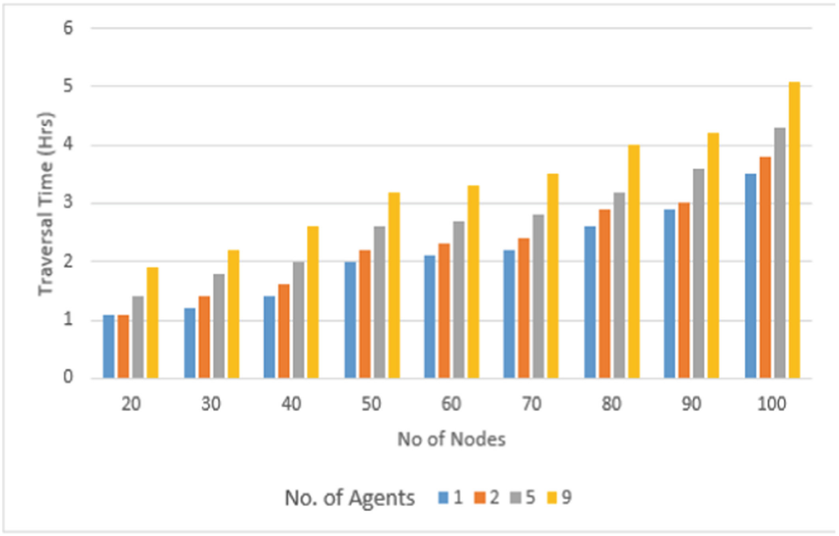

Fig. 2. Total cost graph

We discovered that the model that included imitation and reinforcement learning outperformed all rival models. The model's overall generalization to multiple agents and larger graph sizes is outstanding. Each traversal's cost is distributed across the agents in a fairly equal manner. The method performs significantly better than the current state-of-the-art, LKH solver. The overall cost increases marginally when the number of agents is increased, demonstrating high scalability. When dealing with more agents, the models trained with reinforcement

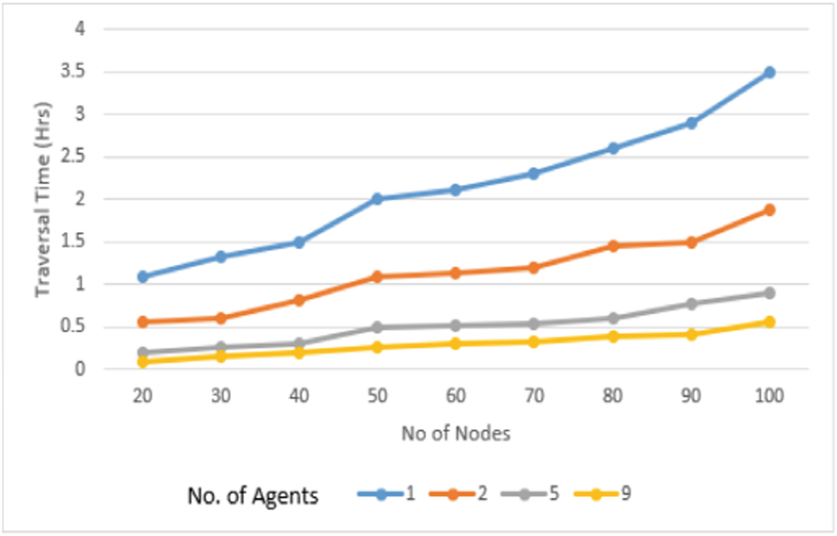

Fig. 3. Cost per agent (runtime in hrs)

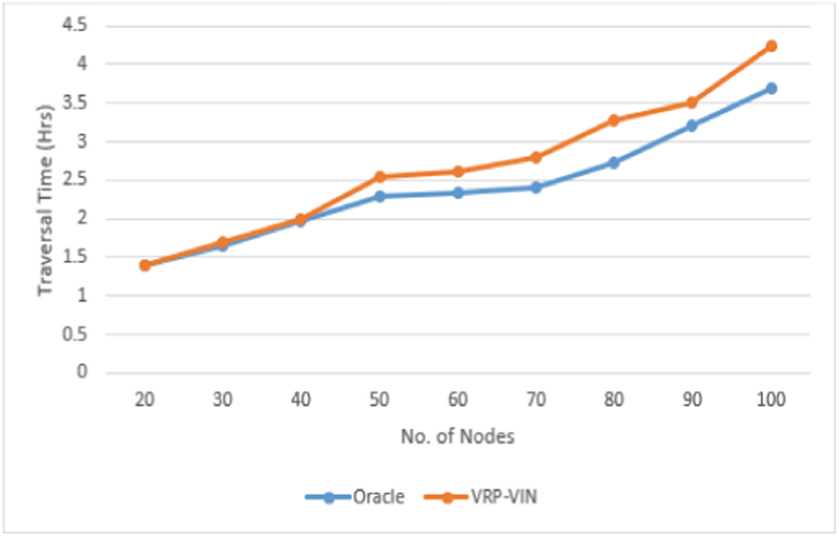

Fig. 4. Oracle vs VRP-VIN (runtime in hrs)

learning have the excellent generalizing capability. Increasing the number of value iterations further increases the performance.

5 Conclusion

The proposed VRP-VIN model can easily route multiple vehicles online in a real-world environment with dynamic obstacles. This model beats all current approaches on real road graphs by leveraging the learned value iteration transitions and a communication protocol based on an attention mechanism. Also, it can be scaled up or down to different numbers of agents and nodes without requiring retraining. Communication is a key component in multi-agent systems learning coordinated behavior. So, Future studies will involve a more in-depth examination of the information stored in the communication and its semantic value. There will also be further research into approaches that will allow this system to operate on huge graphs.

References

1. Pflueger, M., Agha, A., Sukhatme, G.S.: Rover-IRL: inverse reinforcement learning with soft value iteration networks for planetary rover path planning. IEEE Robot. Autom. Lett. **4**(2), 1387–1394 (2019)
2. Toth, P., Vigo, D.: The vehicle routing problem. SIAM, Toronto (2002)
3. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **32**(1), 4–24 (2019)
4. Tampere, C.M., Corthout, R., Cattrysse, D., Immers, L.H.: A generic class of first order node models for dynamic macroscopic simulation of traffic flows. Transp. Res. Part B: Methodol. **45**(1), 289–309 (2011)
5. Sewall, J., Wilkie, D., Merrell, P., Lin, M.C.: Continuum traffic simulation. Comput. Graph Forum **29**(2), 439–448 (2010)
6. Singh, J., Dhurandher, S.K., Woungang, I.: Multi-agent reinforcement learning based efficient routing in opportunistic networks. In: 2020 IEEE 17th India Council International Conference (INDICON), New Delhi, December 2021, pp. 1–6. IEEE
7. Kool, W., van Hoof, H., Welling, M.: Attention, learn to solve routing problems!. In: 7th International Conference on Learning Representations, ICLR, New Orleans, LA, USA (2019)
8. Tamar, A., Levine, S., Abbeel, P., Wu, Y., Thomas, G.: Value iteration networks. In: Advances in Neural Information Processing Systems, NIPS, Barcelona, Spain, vol. 29, pp. 2154–2162 (2016)
9. Lu, J., Li, J., Yuan, Q., Chen, B.: A multi-vehicle cooperative routing method based on evolutionary game theory. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), Auckland, New Zealand, pp. 987–994 (2019)
10. Tang, B., Li, Z., Han, K.: Multi-agent reinforcement learning for mobile crowdsensing systems with dedicated vehicles on road networks. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, pp. 3584–3589 (2021)
11. Niu, Y., Paleja, R., Gombolay, M.: Multi-agent graph attention communication and teaming. In: Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, IFAAMAS, pp 964–973 (2021)

12. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. In: 6th International Conference on Learning Representations, ICLR, Vancouver, Canada (2018)
13. Li, J., et al.: A traffic prediction enabled double rewarded value iteration network for route planning. IEEE Trans. Veh. Technol. **68**(5), 4170–4181 (2019)
14. Sykora, Q., Ren, M., Urtasun, R.: Multi-agent routing value iteration network. In: International Conference on Machine Learning, pp. 9300–9310, November 2020
15. Yun, S., Jeong, M., Kim, R., Kang, J., Kim, H.J.: Graph transformer networks. In: Advances in Neural Information Processing Systems 32, NeurIPS, Vancouver, Canada, pp. 11983–11993 (2019)
16. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, NIPS, Long Beach, CA, USA, vol. 30, pp. 6000–6010 (2017)
17. Dhurandher, S.K., Singh, J., Nicopolitidis, P., Kumar, R., Gupta, G.: A blockchain-based secure routing protocol for opportunistic networks. J. Ambient. Intell. Humaniz. Comput. **13**(4), 2191–2203 (2022). https://doi.org/10.1007/s12652-021-02981-9
18. William, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn. **8**(3–4), 229–256 (1992)
19. Helsgaun, K.: An extension of the Lin-Kernighan-Helsgaun TSP solver for constrained traveling salesman and vehicle routing problems. Roskilde University, Roskilde (2017)
20. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
21. Jiang, J., Lu, Z.: Learning attentional communication for multi-agent cooperation. In: Advances in Neural Information Processing Systems, NeurIPS, Monteral, Canada, vol. 31, pp. 7265–7275 (2018)
22. Erdogan, G.: An open source spreadsheet solver for vehicle routing problems. Comput. OR **84**, 62–72 (2017)
23. Gupta, J.K., Egorov, M., Kochenderfer, M.: Cooperative multi-agent control using deep reinforcement learning. In: International Conference on Autonomous Agents and Multiagent Systems, AAMAS, São Paulo, Brazil, pp. 66–83 (2017)
24. Gupta, N., Singh, J., Dhurandher, S.K., Han, Z.: Contract theory based incentive design mechanism for opportunistic IoT networks. IEEE Internet Things J., 1–11 (2021)
25. Issariyakul, T., Hossain, E.: Introduction to Network Simulator 2 (NS2). In: Issariyakul, T., Hossain, E. (eds.) Introduction to Network Simulator NS2, pp. 1–18. Springer, Boston (2009). https://doi.org/10.1007/978-0-387-71760-9_2