UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft
Faculty for Computer Science, Electrical Engineering and Mathematics

Final Report

Routing in Wireless Network with Reinforcement Learning

Supervisor:
Haitham Afifi

Group Members:
Chethan Lokesh Mariyaklla (6896956)
Pavitra Gudimani (6880555)
Priyanka Giri (6882359)

Paderborn, March 19, 2021

Contents

1. Introduction

Over the years, wireless communication has been achieving exponential growth by making advances in communication infrastructures. The advances made have led to ease of using communication devices such as portable laptops and phones etc. One of the primary attractions of wireless infrastructure is that it does not require fixed wires or cables to transmit the data from one point to another. Today, the increased wireless devices generate a massive amount of data, and it is necessary to route the data to a destination without losing it. To handle this huge data, wireless network infrastructures have to work coexisting. As shown in figure 1.1, the nodes in collision domain 1 are present within one range and are connected to each other. Similarly, nodes in collision domain 2. Nodes present in the overlapping region are intermediate nodes and act as a bridge to transfer packets from collision domain 1 to collision domain 2. Although wireless communication has many advantages, it has its equal share of challenges, which has to be considered while designing the infrastructure. Few important challenges in most of the wireless communication protocols are signal interference, hidden terminal problems and presence of malfunctioning node in the network.

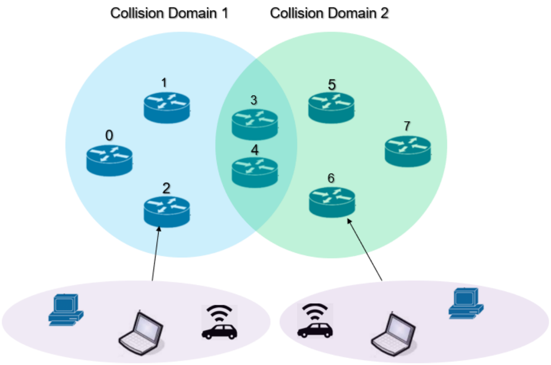

Figure 1.1: Wireless Network

Signal Interference: If more than one node within the same collision domain tries to transmit simultaneously, it causes a collision and leads to packets loss. To avoid the collision, if one node is transmitting a packet at any time, other nodes should refrain from transmitting the packet. As more collisions lead to multiple packet loss, it is essential to coordinate among them to increase a network's efficiency.
For example - in figure 1.2, 2 collision domain network topology, nodes (0, 1, 2, 3,

4) are in the same collision domain. If nodes 0 and 1 are transmitting a packet at the same time, packet collision will occur.

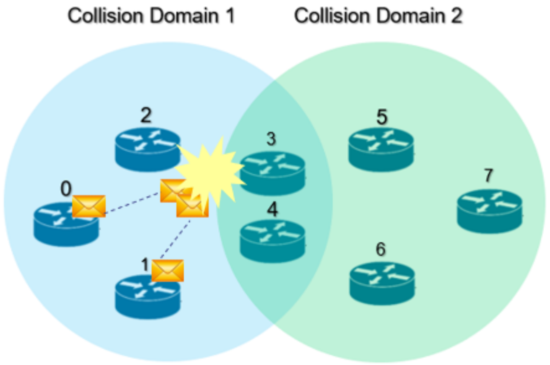

Figure 1.2: Packet collision due to interference

Hidden Terminal Problem: Hidden nodes in the wireless networks are those, which are out of the collision domain of other nodes in the network. In a wireless network, the hidden terminal problem occurs when nodes that are present in different collision domains are communicating with the common node between them. In figure 1.3, node 1 is present in collision domain 1, and node 7 is present in collision domain 2, unaware of each other's transmission. In this situation, they transmit the packet to node 3, a common node for both the collision domain. This action would lead to packet loss at the intermediate node 3 [2].

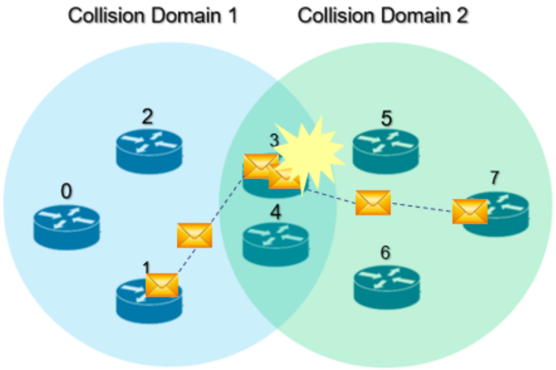

Figure 1.3: Packet collision due to hidden terminal problem

Defect Nodes: A defect node in the network can be any node that is not performing intended action due to malicious attacks or low battery. If a packet is transmitted to this node, it will not be processed and will be considered as lost. In the figure 1.4, node 4 is the defect node. If a packet from source node 1 to destination node 6 is transmitted through intermediate node 4, it will be lost.

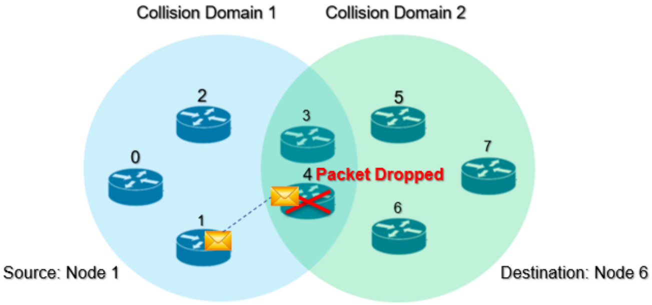

Figure 1.4: Packet collision due to defect node

To reduce the loss of data due to the above challenges, the nodes need to decide when and whom to send the data. So that data is reached to the destination without being lost. In our work, we introduce the Reinforcement Learning (RL) mechanism into wireless networks to explore the possibility of reaching better performance than existing protocols. We propose a centralized RL agent approach for the whole network, where one single RL agent will take actions for all the nodes by observing the environment. And also decentralized approach, in which each node in the network can decide routing and time-slot to send the data. In this report, we will present our problem setup of the above proposed approaches and its performance compared to baseline protocols of wireless networks.

2. Overview of RL

One of AI's primary goals is to generate autonomous agent, which learns through interactions with the environment without any prior knowledge. RL is one of the autonomous agents that learns through continuous interactions with the environment and improves its ability to act over time. RL agent will get the current state of the environment. The agent will perform an action on the environment and receives feedback. Upon receiving the feedback on actions' consequences, the agent can alter its behavior to gain positive feedback. The basic model of RL agent is given in figure 2.1 Initially, at time t, the agent gets an initial state s_t and performs an action a_t. When action a_t is performed, the state s_t will be changed into s_{t+1}. Further, an agent receives the reward r_t for the action a_t and following state information s_{t+1} [7].

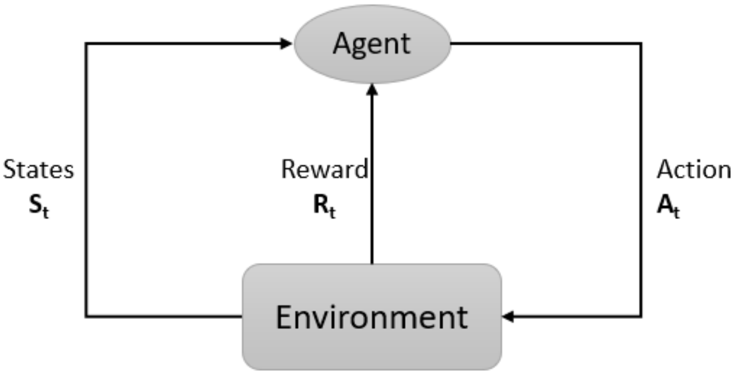

Figure 2.1: Reinforcement Learning Model

3. Problem Statement

Our work mainly aims to evaluate Reinforcement Learning algorithms to route the packets in the network to there respective destination by minimizing the loss of packets due the challenges present in section 1.

4. Environment Setup and Solutions

This section includes a detailed description of the environment on which experiments are conducted, and different solution approaches used to solve the problem using this environment along with the baseline approaches used for performance comparisons.

4.1 Environment

We have experimented with centralized and decentralized agent approach on a network topology with two collision domains, a total of six nodes with two intermediate nodes, and one randomly selected defect node among them.

The figure 4.1 describes our network topology. It involves 6 nodes and 2 collision domains. Nodes (0, 1, 2, 3) are in collision domain 1, and nodes (2, 3, 4, 5) are in collision domain 2. All the nodes within collision domain 1 are connected to each other, similarly in collision domain 2. Nodes 2 and 3 will act as an intermediate node between the nodes which are not reachable from collision domain 1 to collision domain 2. For example - node 0 is not directly connected to node 4. If there is a packet at 0 destined to 4, it takes a multi-hop path to reach node 4 (path taken is 0 to 2, 2 to 4 or 0 to 3, 3 to 4).

A single defect node is chosen randomly for each episode. Each node will have a packet queue with 5 packets except the attack node. Each of these packets will have the destination information. Nodes can be a source, if it is sending the packet. The node can be a destination, if it is receiving the packet. The defect node will neither be considered as source nor destination.

4.2 Solution Approaches

We describe the different baseline protocols, RL agent approaches in this section. Before going into the solution, it is required to understand the Medium Access

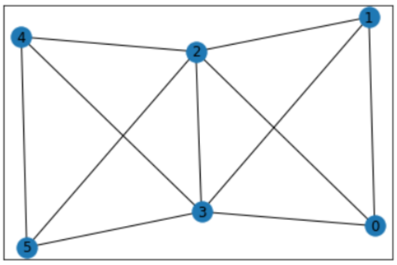

Figure 4.1: 2 Collision Domain Wireless Network Topology

Control (MAC) and Routing mechanism.

Medium Access Control (MAC): In a wireless network, the communication channel is shared among the devices in a network. The nodes in the network may compete for the channel to perform transmission. It is essential to provide an efficient and equal allocation of communication channel among competing users. That is where MAC protocols help in controlling access to the medium. Time-Division Multiple Access (TDMA) is one of the MAC protocols, in which a dedicated time slot is allocated to each node for transmitting the data.[2][6].

Routing: Routing is a mechanism of finding a path from source to any destinations in the network. Many routing protocols are available which enable the routing path search process in the network and store the routing table at nodes. Destination Sequenced Distance Vector (DSDV) is one of the proactive routing protocols. Proactive routing protocols will periodically collect any property changes in the network and update the routing table. [2][4].

4.2.1 Baselines

1. DSDV with priority based round-robin MAC
2. DSDV with round-robin MAC
3. Random agent

4.2.1.1 DSDV with Priority Based Round-Robin MAC

In this approach routing is considered based on DSDV and the time slot allocation for the transferring the packets is done based on round robin mechanism and nodes with highest packets in queue are given priority.

Each node will maintain a routing table with a destination, next-hop and hop-count entries for the destinations, and node ID number in this baseline. The defect node is excluded from the routing table. Initially, each node will store the immediate next neighbors' details and broadcast the routing table to neighbor nodes. Then, nodes will store the received routing table in a queue to add the latest entries into their routing table. As long as there are newly received routing tables in the queue, they will perform the routing table updates. Once they update the routing table, each node will again broadcast the new routing table to their neighbor nodes. The broadcast will happen until each node in the network has all possible destinations in the routing table and stops if there are no routing tables in any node's queue.

Suppose the received routing table has the duplicate entries as already available entries at a node. In that case, the update will happen only if there is less hop-count to any destination in the received routing table. A new routing table will be constructed if they find a new defect node in the network and the process continues further.

Once the routing table is constructed, nodes will transmit the packets based on the number of packets to be transmitted at each node. The node with the maximum packets to transmit will get the priority. If two nodes have the same packets, then the node with the least node ID will be given priority.

4.2.1.2 DSDV with Round-Robin MAC

In this baseline, the DSDV protocol will work in the same as described in the above section 4.2.1.1. The MAC implemented using the round-robin technique. The chance to transmit the packets will be given in sequential order, and once it reaches the last node in the network, it will again start from the first node.

4.2.1.3 Random agent

The approach of this agent is to take random actions to transfer the packets in each time step.

4.2.2 RL Based Approaches

In this section we present approaches using reinforcement learning methods.

4.2.2.1 Centralized Approach without Routing

In this approach, the single RL agent is used to decide for each node in the network when to perform transmission and wait. The agents do not provide the routing path. This approach will already have the routing table constructed using the DSDV routing protocol.

The agent's state will have next-hop for each first packet's destination in each node's queue, maximum queue length a node can have in transmission stage, and defect node information. An action given by the agent will have the transmit or wait decision. The agent's reward system has a negative reward for the actions which cause interference due to two or more nodes within the same collision domain performing transmit action. Furthermore, other actions which lead to a hidden terminal problem or nodes being idle will also have a negative reward. A positive reward is given when a packet is successfully transmitted to a destination.

State: MultiDiscrete [7, 7, 7, 7, 7, 7, 25, 25, 25, 25, 25, 25, 6], indicates the initial [7, 7, 7, 7, 7, 7] are next-hop details for the packet at the source. Next-hop is fetched from DSDV routing mechanism. 25's indicates the maximum number of packets a node can store if all packets are being transmitted to single node. The last digit 6 in the state-space indicate the attack node, it can be any node between 0 to 5.

For example the state returned is [2, 3, 4, 4, 5, 6, 5, 5, 5, 5, 5, 0, 5]. Node 0 has packet whose next-hop is 2, for node 1's packet next-hop is 3. Similarly for node 2, 3, and 4 next-hop is 4, 4, 5 respectively. The last element in the state-space is defect node with no packets, hence no next-hop information is given. The 6 for defect node indicates empty queue at 5. The next elements after 6 indicates the queue size at each of the node in the network. The defect node's queue size is 0.

Action: MultiDiscrete [2, 2, 2, 2, 2, 2], which indicates transmit or wait action. If the action returned by the agent is 1 then it is to transmit. If the action returned by the agent is 0 then it is to wait. For example, the action returned by the agent is [1, 0, 1, 0, 1, 0], then nodes 0, 2, 4 will transmit, and nodes 1, 3, 5 will wait for their turn to transfer.

Reward: Successful transmission to destination gives the maximum positive reward = 1, the transmission via multi-hop path is given a less negative reward = $-\big(0.1\ \times hopcounts\big)$. Packet loss due to collision = -1, packet loss due to hidden terminal problem = -1. If there is no action to transmit for any node in the network, then reward = -1. Agent will terminate the episode when time-steps taken are greater than 600 to transfer, but packets are still left to transfer in the queue or all the packets have been successfully transmitted. If an episode terminates even though queues at any node in the network has packets to transmit, then reward is = -600.

4.2.2.2 Centralized Approach with Routing

In the first approach, the action for each node is to either transmit or wait. In this approach, the RL agent will return transmit or wait action along with the routing path. That is, the RL agent will also provide the next-hop details to each node.

The agent's state will have each first packet's destination details in each node's queue and defect node information. An action given by the agent will have the next-hop information and when to perform transmission. The agent's Reward System is normalized by punishing the agent with a negative reward of -1 for the bad actions that lead to packet loss and an award with positive value of +1 for good action. The bad action includes the agent may ask a node to transmit a packet to defect nodes, or it can cause a collision by allowing two nodes in the same collision domain to transmit a packet at the same time or might lead to a hidden terminal problem or all the nodes are idle.

PPO2 algorithm from stable baseline library was used for this approach.

State: State - Space for the centralized agent for the network topology in figure 4.1 is MultiDiscrete ([7, 7, 7, 7, 7, 7, 25, 25, 25, 25, 25, 25, 6]) - Explanation for state space is as follows, each of the 7 in the array indicates that each node in the topology has freedom of selecting any number from [0, 1, 2, 3, 4, 5, 6]. Since we have only 5 nodes [0 to 5] in the topology, each node can select any number from 0 to 5 as a destination. 25's indicates the maximum number of packets a node can store if all packets are being transmitted to single node. The additional number 6 in [0, 1, 2, 3, 4, 5, 6] indicates the node is empty and no packets to transfer. Furthermore, in state space ([7, 7, 7, 7, 7, 7, 25, 25, 25, 25, 25, 25, 6]), the last element 6 is to indicate the random attack node to be selected from 0 to 5.

For example, the state received by an agent is [1, 2, 3, 4, 0, 6, 5, 5, 5, 4, 5, 0, 5], which indicates node 0 has a packet destined to 1, node 1 has a packet destined to 2, node 2 has a packet destined to 3, node 3 has a packet destined to 4, similarly node 4 to node 0. Node 5 is the defect node with no packets that is why it is 6. The next 5, 5, 5, 4, 5, 0 indicate the number of packets in the queue. The last element in the array 5 is defect node.

Action: Action Space for the network topology in figure 4.1 is MultiDiscrete ([4, 4, 6, 6, 4, 4]) - where the values in each element indicates the nodes present in its collision domain for which it can transfer the packets i.e next-hop options for each node and transmit or wait decision. If action is to transmit to self then that action is considered as wait state. For node 0 and 1, the available options are [0, 1, 2, 3]. For intermediate nodes 2 and 3 available options are [0, 1, 2, 3, 4, 5]. Similarly, for nodes 4 and 5 available actions are [2, 3, 4, 5]. If the next-hop returned by an agent for 0 is one among 1, 2, or 3, the packet will be transmitted to it. If the agent returns 0 that indicates node 0 has to wait.

For example - the action returned by an agent is [1, 1, 2, 3, 3, 3], which tells, next-hop for node 0 is 1, and it can transmit. The next-hop for node 1 is 1, and it should wait. Similarly, nodes 2, 3, and 5 should wait. For node 4 next-hop is 5, and it can transmit.

Reward: Positive reward for successful packet transmission to destination is = 1. Packet transmission to defect node = -1, packet lost due to collision = -1, packet lost due to hidden terminal problem = -1, packets traveling via multi-hop path = -$\left(0.1 \times hopcounts\right)$. If there is no action to transmit for any node in the network, then reward = -1. Agent will terminate the episode when time-steps taken are greater than 600 to transfer, but packets are still left to transfer in the queue or all the packets have been successfully transmitted. If an episode terminates even though queues at any node in the network has packets to transmit, then reward is = -600.

4.2.2.3 Decentralized Approach

As the centralized approach offers limited scalability, as an alternate, we are proposing a decentralized approach. In the decentralized approach, each node acts as an agent will decide independently to transfer the packet, including the next hop, and whether to transfer or not in the current time-step.

The reward system is the same as the centralized environment mentioned in the section 4.2.2.2. The state space is same for all the agents, which is defined as MultiDiscrete ([7, 7, 7, 7, 7, 7, 30, 30, 30, 30, 30, 30, 6]), where 30 is the maximum number of packets present in the network. The action space of each agent is total number of nodes belonging to its collision domain. For example, with reference to network topology in figure 4.1; Node 0 will action space as Discrete(4) as only 0, 1, 2, 3 are present in its collision domain. But, intermediate node like 2 will have action space of Discrete(6), which includes nodes belonging to both collision domain. As already discussed in centralized approach, if a node is transmitting to itself it is considered as wait for that time-step

PPO2 from RLlib library was used for implementing decentalized solution.

5. Results

Performance of the trained RL agents with baseline agents are measured using different metrics such as

- **Packets delivery rate:** Transferring the packets from source to destination. The source and destination can be in different collision domains, hence multi-hop transfers might be required to reach from source to destination.
- **Successful transmission rate:** Capability of the agent to identify the collision-free time slot to transfer the packet to the node which is present in the same collision domain with single hop.
- **Time steps for all packets:** Total time taken for the agent to transfer all the packets in the network to respective destinations. This metric includes the time-steps for both successful and non-successful transmission.
- **Time steps for fixed packets delivery:** Total time taken for the agent to transfer fixed number of packets in the network to respective destinations. This metric includes only time steps for successful packet transmission.
- **Scaling:** Evaluation of agents behaviour with increase in the number of nodes in network.

The evaluation of all the agents were done over 5000 episodes for all the metrics except scaling.

5.1 Packet Delivery Rate

In wireless transmission, one of the important factor of evaluation for a routing protocol is its capability to transfer the packets to its destination successfully. Loss of packets while transferring means loss of information resulting in bad quality of data received at the destination. Baseline agents implemented with DSDV variants transfers all the packets, even the RL agents shown in figure 5.1 RL-MAC (PPO2 agent for centralized agent without routing), and PPO2, A2C (centralized agents with routing) successfully transfers all the packets with occasional loss of few packets. The reason for losing packets occasionally is because it is quite hard to find the global minima while training the RL model. It requires very accurate precision of the hyper parameters used while training and RL behaviour is very sensitive to each hyper parameter, even to the change of seed of random number used. With our experiments we were able to fine tune to get best behaviour of PPO2 agent for most number of the cases. Similarly, with fine tuning we were able to obtain Multi agent performance to transfer 92% of the packets, which is much better when compared to the performance of random agent that is 8%.

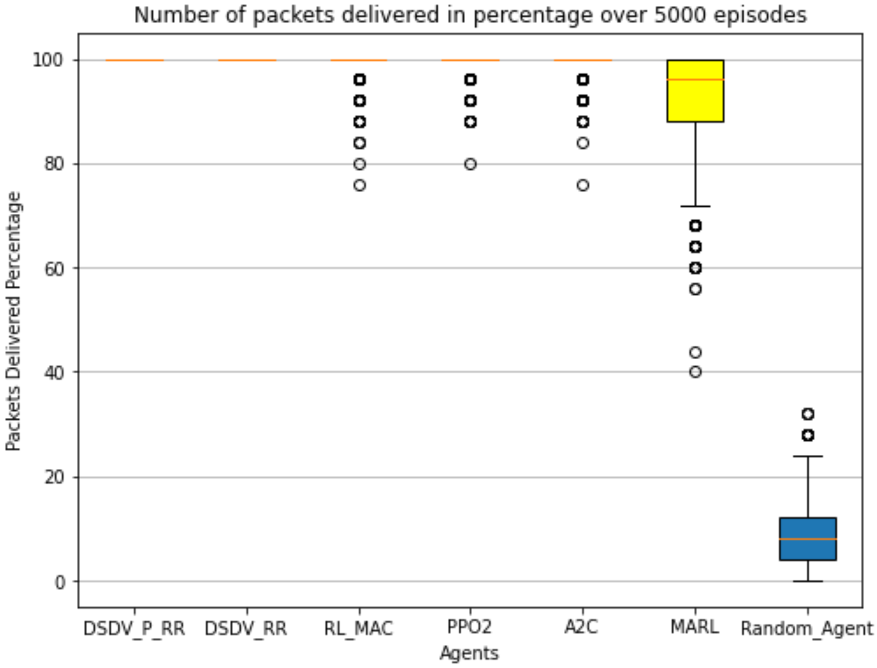

Figure 5.1: Percentage of packet delivered to destination

5.2 Successful Transmission Rate

One of the goals of the RL agent is to find collision free slot for each node to successfully transmit the packet to node in its range. This metric is the measure of MAC protocol implementation. Figure 5.2 shows the box plot for the performance of different agents discussed in section 4.2. As expected, baseline agents implemented with DSDV as routing protocol, round robin and priority based round robin as MAC protocol gives 100% successful transmission as each node is allocated a dedicated time slot for the transfer of packet. Random agent gives the worst successful transmission rate, because of random actions. Trained centralized RL agents shows 100% performance comparable to baseline agents, but there are few instances where the agent makes the mistakes leading to collision. Multi agent RL gives average successful transmission rate of 96%, 85% better than the random agent, but it is 4% behind performance of PPO2 and DSDV baseline variants.

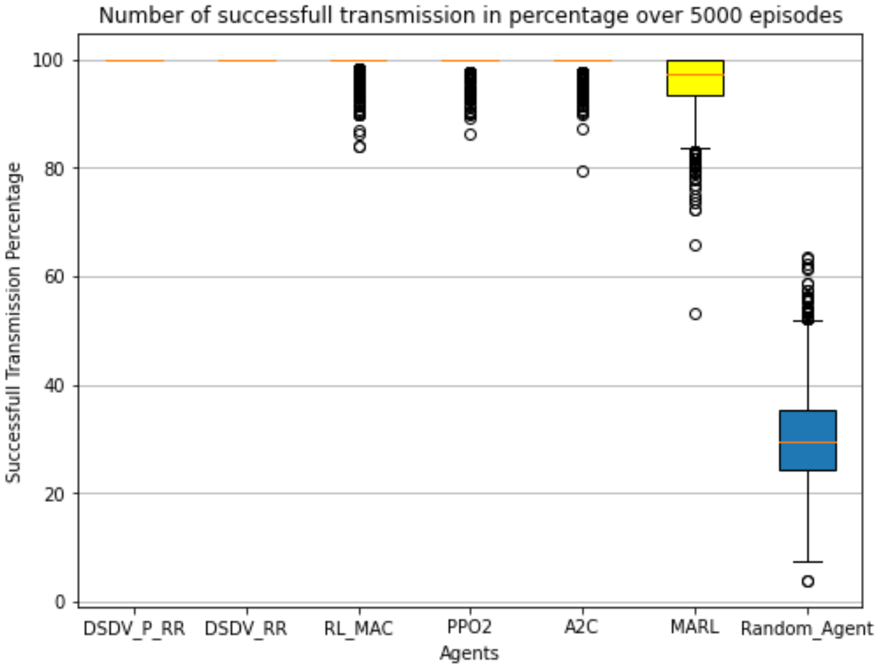

Figure 5.2: Percentage of successful transmissions

5.3 Time steps for all packets

This metric gives us information on total time taken for all the packets that is transferred to the destination. This metric needs to be considered with the packet delivery rate as shown in figure 5.1. Figure 5.3 shows the result for transferring 25 packets, RL agents even though gives occasional packet losses gives comparable results to baselines agents in general. DSDV agent with priority based round robin

MAC protocol which takes 25-30 time-steps, where as RL agents with routing (PPO2 and A2C) takes even lesser time-steps and gives the best performance. Centralized agent without routing and decentralized RL model on an average takes 30-40 timesteps and this is much better performance compared to that of DSDV with round robin MAC protocol which takes 65-80 time steps. Multi agent RL model performs better when compared to that of the random agent as it transfers more packets in less time steps.

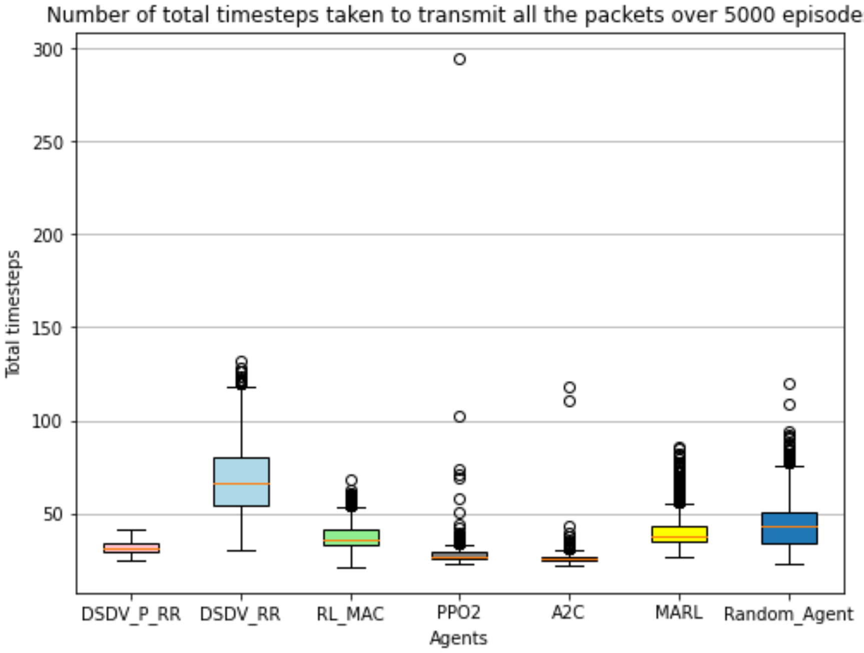

Figure 5.3: Total time steps taken to transmit all the packets

5.4 Time steps for fixed packets delivery

This metric gives us information on how efficiently the agent transfers the packet. We have calculated the time taken by each agent to transfer 15 packets. DSDV variants take 15 time steps to transfer the packets. RL based agents gives better performances as they are capable of transferring packets efficiently by taking less time steps as shown in figure 5.4. This is because RL agents will be able to successfully transfer packet in different collision domains simultaneously where as in DSDV based approaches only one dedicated time slot is assigned for one node at a time. A2C approach shows better performance than all other agents with more number of instances 13-14 time-steps as seen in the box plot, indicating more efficient transfer of the packets in the network.

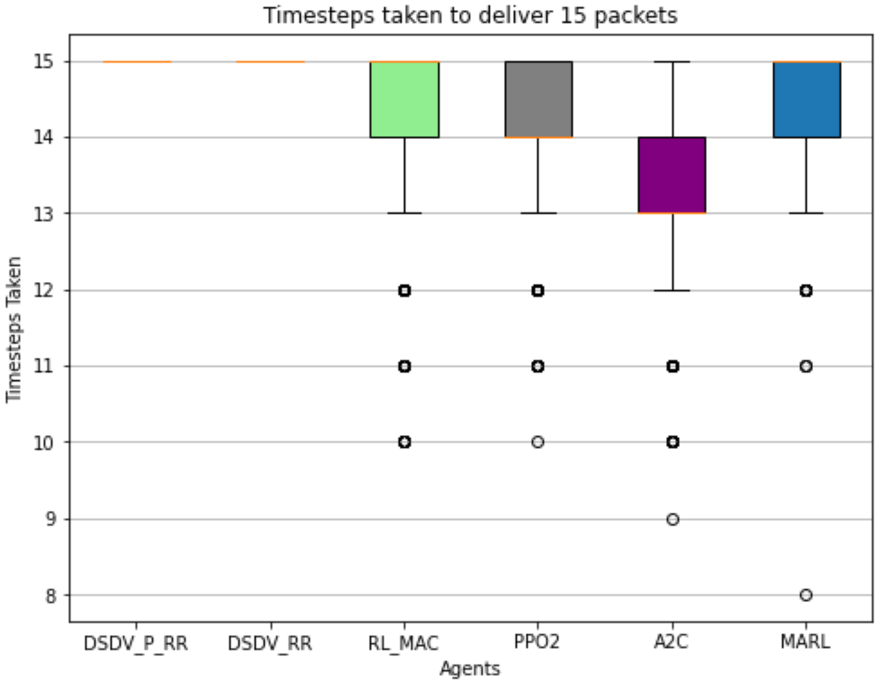

Figure 5.4: Time steps taken to successfully transmit 15 packets

5.5 Scaling Results

In this section we will present the evaluation of centralized and decentralized reinforcement learning agents behaviour with respect to increase in number of nodes. We have considered three network topologies as shown in figure 5.8. While evaluating for this results, network topologies with two collision domains are considered. Increase in number of nodes increase the size of state space and action space which the agent needs to explore before converging to the certain behaviour. This increase is significantly large in case of centralized agents compared to decentralized agents. For example, the state space and action space for network with 8 nodes (5.6) is Multidiscrete([9,9,9,9,9,9,9,9,35,35,35,35,35,35,35,35,8]) and Multidiscrete([5,5,5,8,8,5,5,5]) respectively. Where as for network with 6 nodes (5.5)is Multidiscrete([7,7,7,7,7,7,25,25,25,25,25,25,6]) and action space is Multidiscrete([4,4,6,6,4,4]). But, in case of decentralized agent though the state space size remains the same the action space size is reduced drastically. The action space

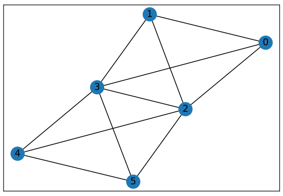

Figure 5.5: Network with six nodes

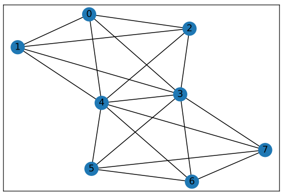

Figure 5.6: Network with eight nodes

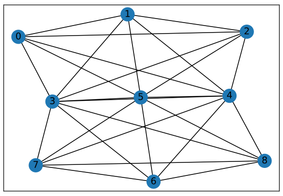

Figure 5.7: Network with nine nodes

needed to be explored by each agent in case of 8 node network is Discrete(8) for intermediate and Discrete(5) for others; for network with 6 nodes Discrete(6) for intermediate and Discrete(4) for other nodes. Hence, we expect the decentralized agents to take lesser time to converge to certain behaviour as the size of network increases when compared to that of the centralized agents.

The metric considered for evaluation of two agents is value function explained variation (VF-EV). This is explained variation of the predicted future rewards obtain from the value functions. The value ranges from 0 to 1 with 1 as the best value it can achieve. Figure 5.8 shows the VF-EV for the centralized agent with 6,8 and 9 nodes. We can observe that with increase in number of nodes, time taken for convergence has also been increased. In figure 5.8 the orange line representing network with 6 nodes, value has converged around 4 million time-steps and 6 million in case of 9 nodes (5.8, light blue line). In case of decentralized, the graph figure 5.9, shows the VF-EV for only one agent. As we can observe, behaviour of this agent has been converged well within in 1.5 million time-steps for all the networks. Also, there is only slight increase with increase in the number of nodes in network. With these results we can infer that decentralized agents are easily scalable when compared to that of centralized agents.

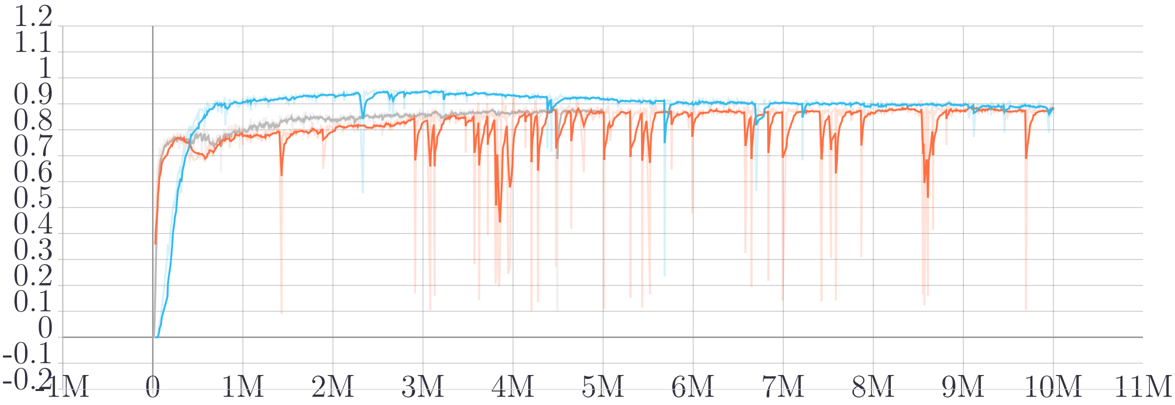

Figure 5.8: Value function explained variation for centralized agents with color representation follows, orange - 6 nodes network, gray - 8 nodes network and light blue - 9 nodes network

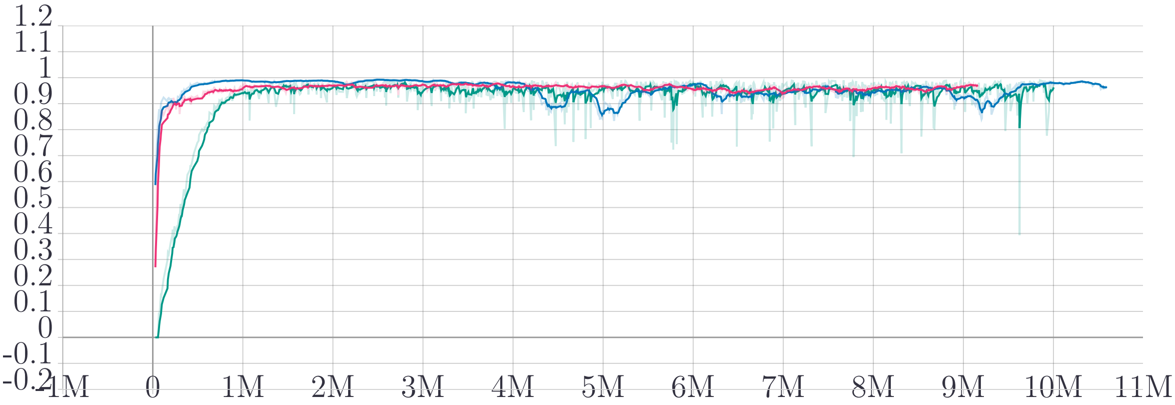

Figure 5.9: Value function explained variation graphs for decentralized agents with color representation as follows, blue - 6 nodes network, pink - 8 nodes network and green - 9 nodes network

6. Conclusion

With our experiments in training reinforcement learning agents to transfer packets in wireless network, we were able to train the PPO2 and A2C network to match the performance of baseline agents. From the results, we observed RL agents were more efficient in transferring the packets by taking less time-steps. Also, with increase in network size we observed that decentralized agent is more scalable when compared to centralized agents.

7. Future Work

In our proposed approaches, we have used User Datagram Protocol (UDP) communication protocol. If the packet is lost in the network, it is not resent. This work can be extended using the Transmission Control Protocol (TCP). The performance of RL agents can be improvised with more hyper-parameter tuning. As a MAC protocol, we are using the TDMA technique. It can also be implemented using probability-based MAC protocols such as CSMA/CA.

8. Lessons learnt, Pitfalls

- We faced a challenge in designing the reward system for the problem statement. Over the period, several attempts were made to achieve the current results by changing the reward system. So, it is crucial to decide the reward system that directs the agent in achieving expected performance.

- The size of the action space and state space also implies the convergence of the agent's performance.

- And the most important is to document all the results, and different approaches tried by the team, which will help in report writing. And have a backup of trained models.

9. Other Explored Approaches

- Before implementing the approaches mentioned in the solutions, we started with a simpler problem statement. We tried with a single collision domain without defect node and simple action space to give transmit or wait action without routing as an initial step.

- Next, we tried with the two collision domain network without defect node and action to transmit or wait. To move towards the problem statement, we added a defect node into the network and tried to achieve better performance.

- We also tried RL agents such as A2C and PPO2. The agent's reward system was modified many times, and to get even better results, we tried to tune the hyper-parameters.

- Different search algorithms and schedulers were tried to get the tuned parameters. Gephi was used to show the simulation of RL agents.

- To measure the performances, we implemented the different baselines. Moreover, to improve the scalability of the network, we decided to use the Multi-Agent RL technique.

- In MARL, various agents were tried to achieve the results. Both RL and MARL approaches were tested on a larger network with 4 collision domains and 12 nodes.

- While implementing probability CSMA/CA in wireless connection we faced problems for agent to decide action space in float values. So we considered using real values from 1 to 10 as probability distribution. Agent actions are suggested only for one time step. It was difficult to decide which state space will give more accurate results. Reward was given as same as in centralized RL agent. It was difficult to figure out packet loss count at some places.

Bibliography

[1] URL: https://www.alexirpan.com/2018/02/14/rl-hard.html.

[2] Falko Dressler. *Self-Organization in Sensor and Actor Networks*. 2007.

[3] X. Guo et al. "Deep-Reinforcement-Learning-Based QoS-Aware Secure Routing for SDN-IoT". In: *IEEE Internet of Things Journal* 7.7 (2020), pp. 6242–6251. DOI: 10.1109/JIOT.2019.2960033.

[4] *Routing*. URL: https://en.wikipedia.org/wiki/Routing.

[5] *Signal Interference*. URL: https://en.wikipedia.org/wiki/Interference_(communication).

[6] *TDMA*. URL: https://en.wikipedia.org/wiki/Time-division_multiple_access.

[7] J. Wang et al. "Smart Resource Allocation for Mobile Edge Computing: A Deep Reinforcement Learning Approach". In: *IEEE Transactions on Emerging Topics in Computing* (2019), pp. 1–1. DOI: 10.1109/TETC.2019.2902661.

[8] Y. Yu, T. Wang, and S. C. Liew. "Deep-Reinforcement Learning Multiple Access for Heterogeneous Wireless Networks". In: *2018 IEEE International Conference on Communications (ICC)*. 2018, pp. 1–7. DOI: 10.1109/ICC.2018.8422168.