SAPIENZA
UNIVERSITÀ DI ROMA

Autonomous Networking a.y. 22-23

Homework 1: Report

Davide Gabrielli - gabrielli.1883616@studenti.uniroma1.it
Hazem Dewidar - dewidar.1883881@studenti.uniroma1.it

January 11, 2023

1 Introduction

This homework requires us to implement the Q-Learning algorithm presented in the paper "A Q-Learning-Based Topology-Aware Routing Protocol for Flying Ad Hoc Networks" by Arafat and Moh. The paper presents a location-based routing which not only utilizes the information of single-hop neighbors but also uses the information of two-hop neighbors (Fig. 1).

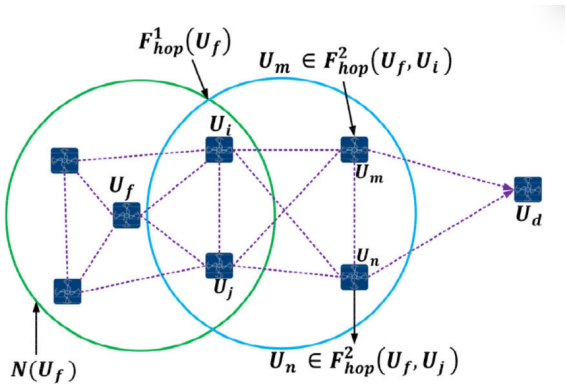

Figure 1: The topology of the network. $\mathbf{U_f}$ is the drone that is currently forwarding the packet. $\mathbf{U_d}$ is the drone that is the destination of the packet. $\mathbf{F^1_{hop}}(\mathbf{x})$ is the set of the one-hop neighbors of drone $\mathbf{x}$. $\mathbf{F^2_{hop}}(\mathbf{x}, \mathbf{y})$ is the set of the two-hop neighbors of drone $\mathbf{x}$ that are one-hop neighbors of drone $\mathbf{y}$. The green circle represents the one-hop neighbors of the drone $\mathbf{U_f}$. The blue circle represents the two-hop neighbors of the drone $\mathbf{U_f}$ (and the drones in between) that aren't one-hop neighbors of the drone $\mathbf{U_f}$.

The main contributions given by the given article are:

- **Topology aware routing**: QTAR can operate in highly dynamic environments. For topology control, two-hop neighbor information is utilized to estimate the environment and node states.

By using two-hop neighbor information, we obtain the node position, link quality, such as delay, velocity, and residual energy level. The two-hop-based geographic routing scheme improves the routing performance and ensures successful forwarding from source to destination

- **Minimazed communication** : the limited battery power of the dones means that the communication delay must be short. So we change dinamically the hello message interval and the link holding timer to minimize the communication delay basing on the link duration when the topology changes.
- **Loop free routing**
- **Adaptive Q-Learning**: The Q-learning method learns from the network environment and allows a quick adaptation.
- **Load Balance**

2 Models

2.1 Q-Learning and UAVs

Q-learning can be used to design intelligent control policies for unmanned aerial vehicles (UAVs), also known as drones. For example, a Q-learning algorithm could be used to teach a drone to navigate through an unknown environment and avoid obstacles. In this case, we traing the model to let the agents learn what are the best drones to relay the packets to based on their two-hop neighbors.

The one-hop neighbors are the drones that are within the communication range of the current drone, and the two-hop neighbors are the drones that are within the communication range of the one-hop neighbors. This allows the drones to make decisions based on the information they have about their neighbors, and the information they have about their neighbors' neighbors thus making the decisions more informed. That's because maybe the best drone to relay to is near the first hop neighbor.

2.2 QTAR

The algorithm proposed in the paper "A Q-Learning-Based Topology-Aware Routing Protocol for Flying Ad Hoc Networks" by Arafat and Moh is called QTAR. QTAR not only utilizes the information of single-hop neighbors but also uses the information of two-hop neighbors.

The authors believe that this method improves the path discovery process, reduces the time required for calculating routes and improves the selection of the next-hop node. Although this technique increases routing overhead and system complexity. QTAR considers information such as delay, speed, and energy when selecting next-hop node. Moreover, QTAR provides a technique for calculating the link lifetime to estimate the Hello interval and the link holding time. The link holding time is In this protocol, Q-Learning factors such as learning rate and reward factor are adaptively determined with regard to network conditions. For simplicity, this implementation uses a 2D environment instead of a 3D environment as proposed in the paper.

The algorithm can be divided into two modules: a module that constructs the topology and a module that selects the best relay and makes decisions based on the qTable.

2.2.1 Topology construction

For a drone to let everyone in the area to know its existance, it need to broadcast hello messages. A drone will also listen for hello messages. Here we have two phases: one hop neighbor discovery and two hop neighbor discovery. At the end of each phase, we update the hello message interval. The hello message interval should be lesser than the link holding timer of the minimum link duration timer.

One hop neighbor discovery For every hello message received, record the presence of originator and calulate the link holding timer and the link duration.

Two hop neighbor discovery Each neighbor in his hello message will also include the list of its one-hop neighbors. For every neighbor included in each hello messages, update the two hop metrics

such as the link duration and the link holding timer taking into account the one hop neighbor.

2.2.2 Routing decisions

To decide the next hop, we will only consider the two-hop neighbors of the drone. We will choose the best two-hop neighbor and select as next hop the drone in between the current drone and the best two-hop neighbor.

If we have the destination in the list of two-hop neighbors, we will choose it as next hop. Otherwise, we will choose the drone that has the highest Q-value. We will then update the Q-value of the drone that has been chosen as next hop. If there was no optimal drone to relay to, we will choose to keep the packet in the drone.

3 Implementation

3.1 Changes to the simulator

For the implementation of the QTAR algorithm, we had to make some changes to the simulator.

3.1.1 Hello Packet

As far as concerns the **Hello Packet**, we made the following changes:

- **Link holding timer**: the time after which a link is considered broken.
- **Sequence number**: the sequence number of the packet. This is used to check if a packet is a duplicate or not. We discarded this part from the paper because it will never happen in this simulator
- **One-hop neighbors**: each drone will attach to the hello packet its list of one-hop neighbors. This is used to update the list of two-hop neighbors of the other drones.
- **Two-hop neighbors**: each drone will attach to the hello packet its list of two-hop neighbors.

3.1.2 Drone

As far as concerns the **Drone**, we made the following changes:

- **Speed**: the speed of the drone. This is used to move the drone in the environment. We modified it making it random at initialization.
- **Hello interval**: the time between two hello packets.
- **Link holding timer**: the time after which a link is considered broken.
- **Distance between the drones**: a list of the distances with the other drones and the time at which the distance was measured.
- **One-hop neighbors**: the list of one-hop neighbors of the drone.
- **Two-hop neighbors**: the list of two-hop neighbors of the drone.
- **Old one-hop neighbors**: the list of one-hop neighbors of the drone at the previous time step.
- **Residual energy**: the residual energy of the drone. The drones will lose energy during the simulation.

3.1.3 Utilities

In the utilities, we added some functions that measure:

- **Delay between two drones**: the time between the moment a packet is sent and the moment it is received. This is 1 second when the drones is at the max distance (distance = communication range) and 0 when the drones is at the min distance (distance = 0).
- **One hop speed**: this is the ratio between the distance from the i-th neighbor drone and the delay.
- **Two hop delay**: this is the ratio between the distance from the i-th neighbor drone and the sum delay of the two path segments (one hop delay and two hop delay).
- **Required speed**: the speed required to reach the destination in the time required to reach the destination.

3.1.4 Routing

We made two classes, AdvancedRouting and QTARRouting, in which the latter inherits the former in order to implement the Algorithm 1 and the Algorithm 2 of the paper. In the AdvancedRouting class, we implemented the exchange of Hello Packets between the drones and the construction of the topology. In the QTARRouting class, we implemented the reward function, the states and actions of the drones (the relay selection algorithm) Furthermore, in the QTARRouting class, we take as action the best drone to relay to.

- **Exchange hello packets**: the drones will exchange hello packets with their neighbors.
- **Process hello packets**: the drones will process the hello packets received from their neighbors.
- **Reward function**: the reward function (Eq. 1) is the same as the one proposed in the paper. The function is defined as:

$$R = A \cdot delay + B \cdot \frac{two\text{-}hop\ node\ speed}{total\ two\text{-}hop\ nodes\ speed} + C \cdot \frac{perc\ residual\ energy}{total\ perc\ residual\ energy} \tag{1}$$

  where:

  delay is the transmission delay between the forwarder and the two-hop neighbor.

  two-hop node speed is the speed of the two-hop neighbor.

  total two-hop nodes speed is the sum of the speed of all the two-hop neighbors.

  perc residual energy is the percentage of the residual energy of the two-hop neighbor ($\frac{residual\ energy}{init\ energy}$).

  Total perc residual energy is the sum of the percentage of the residual energy of all the two-hop neighbors.

- **States and actions**: the states and actions are the same as the ones proposed in the paper.

3.2 Results

In this section, we will show the results of the implementation of the QTAR algorithm compared to the other algorithms (Georouting, Random, and our best Q-Learning algorithm, CDC-GEO, from the previous homework). For each algorithm, we will show the average delivery ratio, the average delivery time, and the average number of relays. For each metric, we will show the average results for the different number of drones (5, 10, . . . , 30) using different seeds (so having different positions and paths for each of them).

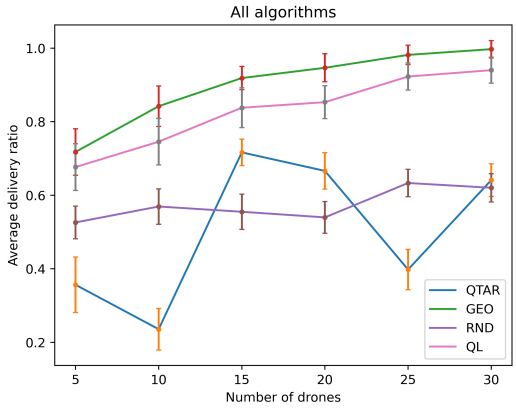

Figure 2: Average delivery ratio for QTAR compared to the other algorithms

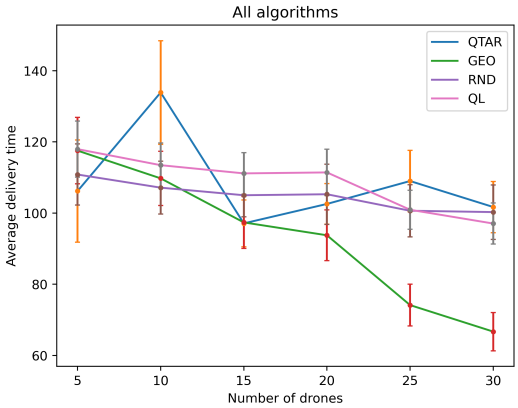

Figure 3: Average delivery time for QTAR compared to the other algorithms

As we can observe, QTAR performances highly depends on the environment setup. In fact, with 15, 20, and 30 drones as we can see from Fig. 2, QTAR performs better than the random baseline. As far as concerned the average delivery time (Fig. 3), with 15 drones, it matches the best algorithm (Georouting).

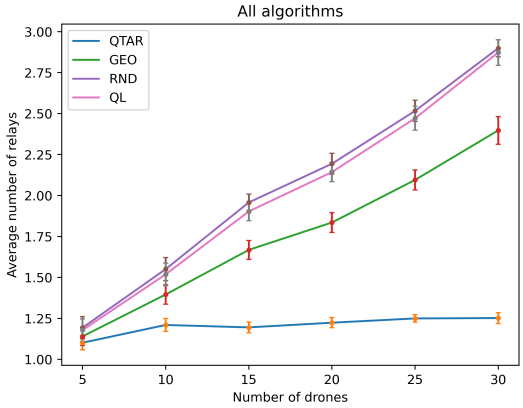

Figure 4: Average number of relays for QTAR compared to the other algorithms

It's interisting to observe that QTAR is the one that relays the packet less frequently when compared to the other algorithms (Fig. 4). Georouting is the one that performs the better, it even archives in some simulation almost a 100% delivery ratio.

4 Conclusions

In this homework, we have implemented the QTAR algorithm and compared it to the algorithms of the previous homework. The results show that the QTAR algorithm is not performing as well as the Q-Learning algorithm or the Georouting algorithm. This is because in our problem we know the exact position of the destination (the depot) and we know that this position won't change during simulation.

QTAR algorithm in fact has been developed to achieve a different task, which is to find the best path to reach a destination drone that is moving. In our case, the depot is static and we know its position, so the QTAR algorithm is not the best choice.

SAPIENZA
UNIVERSITÀ DI ROMA

Contributions

Each student in the group is required to state their contribution to the homework.

Davide Gabrielli Paper Analysis, Algorithm 2 (paper), Plots, Simulator Adaptation

Hazem Dewidar Paper Analysis, Algorithm 1 (paper), Plots Analysis