Multi-agent Deep Reinforcement Learning (MADRL)-Enabled Task Offloading and Resource Allocation in IoT-Edge-Fog Networks

Endris M Ali, Jemal H Abawajy, Muhammad Bilal, Frezewd Lemma, and Mohammed Seid

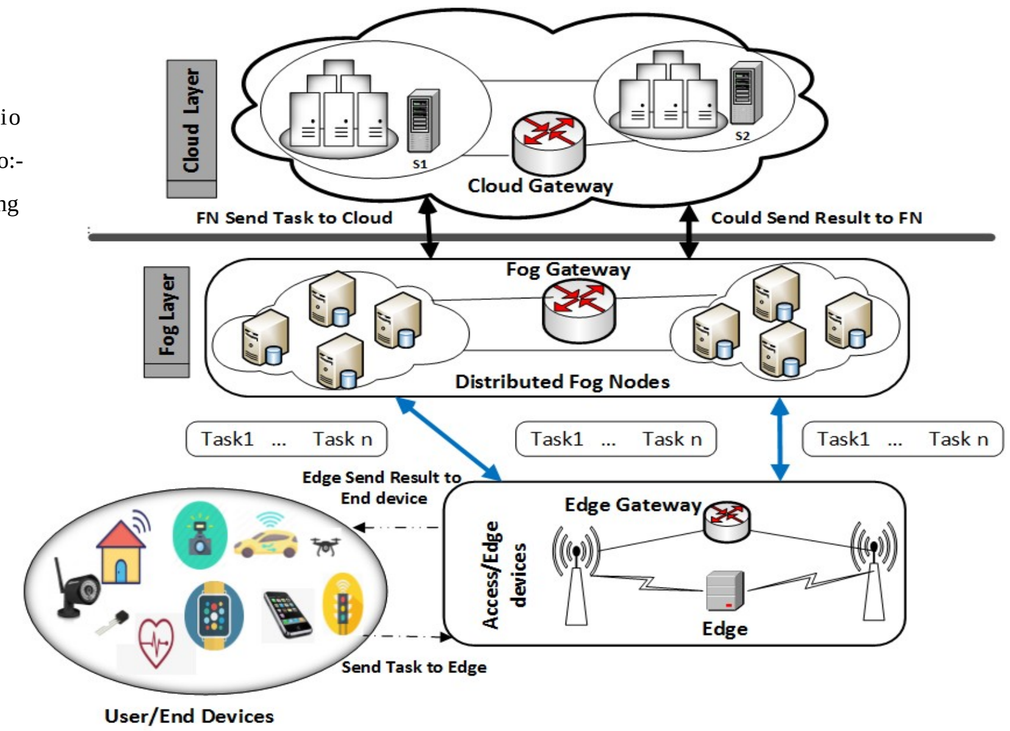
Objective

- The objective is to design a Mult-Agent DRL based Task Offloading and Resource Allocatio Approach for Fog Computing Environment to:-
  - Optimize task offloading decision with minimizing the cost of computation delay and energy consumption to improve the delay critical QoS requirements for real time applications
- We consider a network and computation model described as a multiple tasks with multi-Fog environments and the complex nature of task requirements in both wired and wireless-based services
- The major focus will be
  - IoT-Edge-Fog specifically between distributed FOG while sending task to neighbor Fog and returning result back

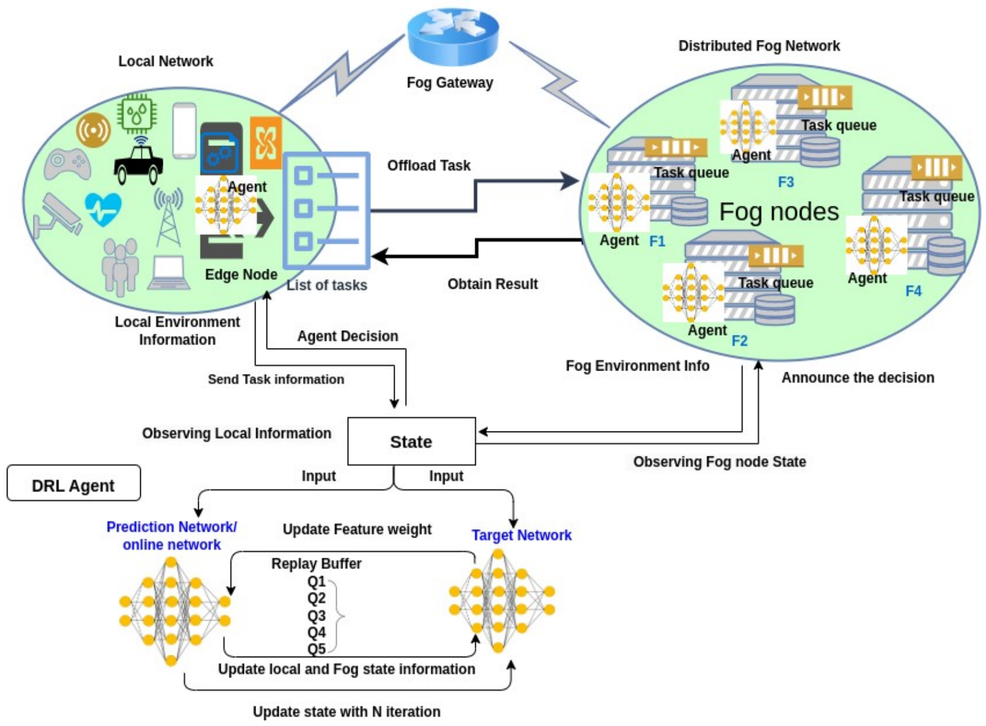
Problem

- The heterogeneous and dynamic nature of the Fog networks
  - Resource status and task allocated in each Fog node varies through time
- These are main concerns in
  - Where to offload,
  - Communication model,
  - Computation model and
  - D/R Learning approaches and extract real-time information from the Fog and Task.
- Generally, our model is allows to do
  - Generate data from the the environment and store it in the replay buffer .
  - Use the data to train the model from the replay buffer
  - Inject the agent with Fog environment and enable with history using Deep Learning
  - Evaluate the result

Overview



Approaches (Methodology)



Algorithm

Algorithm 1 DRL (DDQN) for Offloading Tasks in a Distributed Fog Computing Environment

Input : K_i , S_i, U, F, $A(t)$, environment action space (E), and reward (φ), number of training episodes
Output: Trained agent's policy and Q-values
Step 1: Define the environments
Tasks to be offloaded $K_1, K_2 ... K_n$, Task Size (TS);
Get initial state representation of F_i be S_i with a transition s', resource ($M_{f,i}$, $C_{f,i}$)), action (A_t), and reward function for the DRL agent (S_t, A_t);
Step 2: Initialize the DRL agent
Initialize the deep neural network architecture for the DRL agent;
Initialize the hyperparameters learning rate $\alpha = 0.1$, discount factor $\gamma = 0.1$,and exploration rate $\epsilon = e^2$;
Step 3: Loop for training
Initialize the environment and reset the state;
for *each time step in the episode* **do**
  Get observation state $E(t)$;
  Choose an action a_i from the action space based on the DRL agent's policy;
  Offload the task K_i to a Fog node according to the chosen action a_i;
  Observe the next state s, reward $R(t)$;
  Store the experience (S, A, R, S') in the replay buffer $\mathcal{D}$;
  if *the train starts then* **then**
    Update the neural network parameters using the $\mathcal{D}$ with Q-Learning;
    Update the exploration and exploitation rate with epsilon-greedy decay;
    if *the episode is finished* **then**
      break out of the time step loop;
    end if
  end if
  Update the target network parameters (if using a target network);
  Repeat the training loop for a specified number of episodes or until convergence;
end for
Step 4: End of algorithm

Conclusions

We deployed a MADRL based task offloading algorithm in the Fog to enhance different delay-sensitive application domains in a real-time manner.

Specifically, techniques for optimizing the cost of task computation delay and energy consumption QoS requirements in task offloading for both dynamic and static topological Fog architectures

Contact:Endris M Ali Email: endris.mohamed@astu.edu.et