Q-Learning in Mobile Edge-Computing

Q-Learning in mobile edge-computing is a reinforcement learning algorithm which trains itself on the basis of parameters provided during environment creation and server allocation strategies. Later the trained model can be used to allocate the tasks on the servers in an optimised manner. In local computing, the tasks offloading latency will be on higher side.In MEC, the state space could represent the current network conditions, device status, available resources (e.g., CPU, memory, bandwidth), and user demands. This state information is crucial for making resource allocation decisions. Q-learning uses a reward mechanism to evaluate the quality of actions taken in a given state. In MEC, rewards can be defined based on various performance metrics, such as latency, energy consumption, throughput, or user satisfaction. Higher rewards are associated with better decisions.

The learning process involves iteratively exploring the state-action space and updating the Q-table based on the rewards received. Q-learning uses the Bellman equation to update Q-values iteratively: P(s, a) = P(s, a) + α \* [R(s, a) + γ \* max(P(s', a')) - P(s, a)], where

`P(s, a)` is the Q-value for state `s` and action `a`.

`α` is the learning rate.

`R(s, a)` is the immediate reward for taking action `a` in state `s`.

`γ` is the discount factor.

`max(Q(s', a'))` represents the maximum Q-value for the next state `s'` and all possible actions `a'`.

Algorithm:

Input : Pt, Pt0 , Pre\_node, Comp\_list, Trans\_amount

Output : Trans\_energy

Initialization : Trans\_energy → 0;

If Pt ≠ 0 and Pre\_node (Pt(0)(0)) ≤ Comp\_list then Trans\_energy += ε ptr

If Trans\_amount ≥ Pt(0)(2) then Pt0.append (Pt(0))

sort tasks on Pt0

else Pt(0)(2) -= Trans\_amount

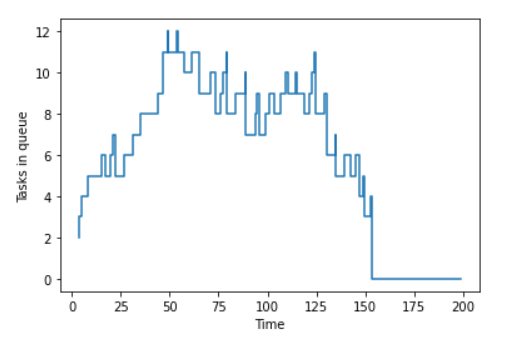


Fig: The above figure shows the number of tasks in queue with respect to time on the basis of provided number of nodes, environment variables, CPU requested and process\_time.

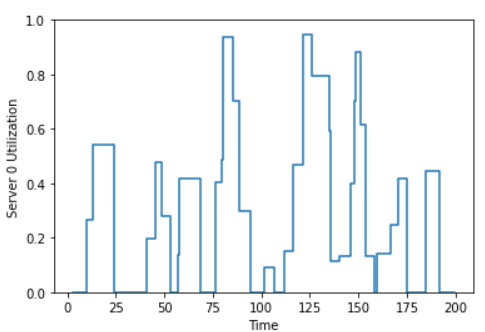
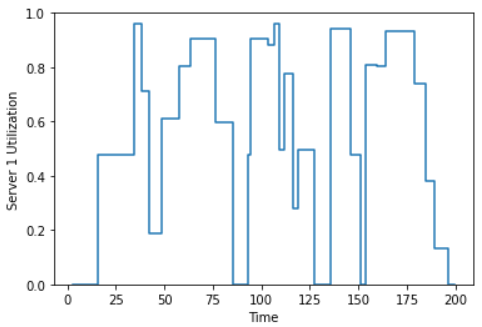
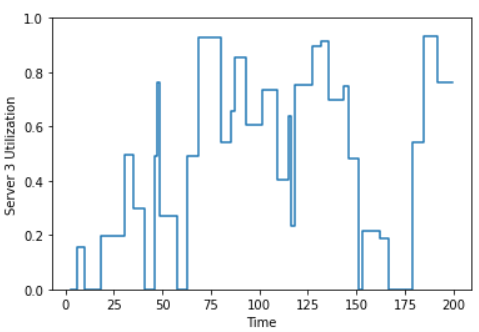
  

Fig: It analyses the server utilization of the three servers taken into consideration and its utilization with respect to time.

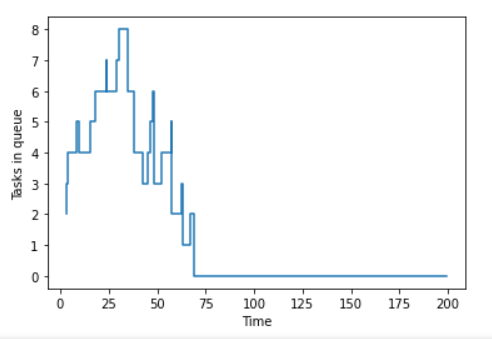


Fig: This figure shows the number of tasks in queue vs the time constrain.

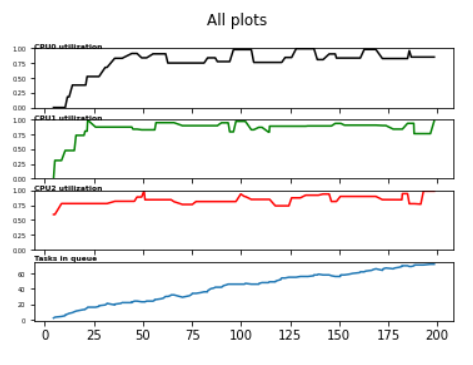


Fig: CPU utilization of different CPU’s and number of tasks in queue during a single window execution