北京邮电大学 本科毕业设计（论文）中期进度报告

Project Mid-term Progress Report

学院 School	International School	**专业 Programme**	**e-Commerce Engineering with Law**		
姓 Family name	Ke	**名 First Name**	Jiaming		
BUPT 学号 BUPT number	2019213499	**QM 学号 QM number**	190894179	**BUPT 学号 BUPT number**	2019213499
论文题目 Project Title	Meta Reinforcement Learning for Task Offloading in MEC Networks				

是否完成任务书中所定的中期目标？Targets met (as set in the Specification)?
YES

已完成工作 Finished work:

1. Simulating a mobile edge computing environment for task offloading
2. Transform the problem of task offloading into a Markov decision process (MDP)

The MEC architecture consists 3 levels, user level, edge level and cloud level. At the user level, there are multiple mobile devices(UE) that generate tasks to be executed. At the edge level, there are multiple edge servers that can accept offloaded tasks from mobile devices and address the latency and bandwidth limitations of cloud computing by bringing the computing resources closer to the users and devices that generate and consume data. The cloud server can also handle tasks that cannot be executed at the edge level due to resource limitations. If task t_i runs locally on the UE, there is only running latency on the UE. The end-to-end latency of a task offloading process includes local processing, uplink, downlink, and remote processing latency. The scheduling plan for a DAG, G = (E, V).

There are numerous MEC networks with diverse mobile devices and edge servers states, and different their available resources, network connectivity, and other relevant parameters. To simulate real task offloading scenarios, We model the dynamic computation offloading process as multiple MDPs of following componednts:

$$1.\quad point\ at\ time\ i.$$

1. States: The state space in the MDP can be defined by the current status of the device and the network, such as the current CPU usage and memory usage, network latency, and available bandwidth. We can use this information to define a set of states that represent different combinations of these factors.

$$S = s_1, s_2, \dots, s_n, where\ \ s_i = \langle u_i, w_i, l_i, b_i \rangle\ is\ a\ state\ that\ represents\ the\ current\ values\ of\ CPU\ usage,\ workload,\ network\ latency, and\ available\ bandwidth, respectively, at\ time\ i$$

2. Actions: The actions in the MDP represent the decision to either offload the task or execute it locally on the mobile device. We can define a set of actions that include "offload" and "execute locally". The action space is defined as $A_i = \{0,1\}$ where A_i = 0,1 indicate that the task is execute locally, or offload to edge at time step i.

3. Rewards: To define the reward function in MDP, we want to consider two objectives: minimizing latency and minimizing energy consumption. We can represent the reward function as a linear combination of these two objectives, where the coefficients determine the relative importance of each objective.
   Let L(s, a) be the latency associated with executing the task according to the chosen action, and E(s, a) be the energy consumption associated with the chosen action. Then, the reward function can be defined as:

$$R(s,a) = -\alpha * L(s,a) - \beta * E(s,a)$$

   where α and β are non-negative coefficients that determine the relative importance of minimizing latency and minimizing energy consumption, respectively. The negative sign indicates that we want to minimize the reward function.
   The reward function can be used to evaluate different policies for task offloading and execution in a dynamic MEC framework. By maximizing the expected cumulative reward over a horizon of T steps, starting from an initial state s0, we can find the policy that optimizes both latency and energy consumption.

Using these state, action, and reward definitions, we can model the task offloading problem as a Markovian optimization problem, and use Meta-RL techniques from Markov decision processes, such as the Q-learning algorithm, Model-Agnostic Meta-Learning or the policy iteration algorithm, to learn an optimal policy for task offloading on each mobile device.

尚需完成的任务 Work to do:

1. Define the characteristics of the MEC environment framework, including the hardware and software configurations of the components at each level, as well as the network topology and traffic patterns. The software configurations can include the operating system, virtualization platform, and containerization platform used on each device.
2. Formulate the optimization objective: choose an appropriate technique to generate the optimal value function or policy.
3. Find the policy that maximizes the expected cumulative reward.

存在问题 Problems:

1. How to fit our MDP models of edge computing in MEC environments into Meta-RL solution.
2. The choice of Meta-RL algorithms.

拟采取的办法 Solutions:

Q-learning involves learning an optimal action-value function, which can be used to determine the best action to take in any given state. One approach to meta reinforcement learning using Q-learning is to use experience replay and dynamically adjust the learning rate during training to improve adaptation to new tasks.

MAML is a model-free algorithm, which works by training a model to learn a good initial set of parameters that can be adapted to new tasks with only a few examples. During the adaptation phase, the model is fine-tuned to the new task by updating the parameters based on the few samples available for the new task. MAML has been shown to be effective in various domains, including image classification, robotic control, and natural language processing.

In general, Q-learning is typically easier to implement than MAML for meta reinforcement learning problems. Q-learning is a model-free algorithm, meaning that it does not require knowledge of the underlying system dynamics and can be applied to a wide range of tasks. On the other hand, MAML is a more complex algorithm that requires the learning of an initial set of parameters that can be adapted quickly to new tasks. This can make it more challenging to implement and fine-tune for specific applications. However, the choice of algorithm ultimately depends on the specific requirements and goals of the problem being addressed.

论文结构 Structure of the final report: (Chapter headings and section sub headings)

Chapter.1- Introduction

- 1.1 Abstract
- 1.2 Background
- 1.3 Related work

Chapter.2- System modelling and Problem formulation

- 2.1 System modelling
- 2.2 Problem formulation

Chapter.3- Implementation of Meta-RL

Chapter.4- Performance evaluation

- 4.1 Results analysis
- 4.2 Model portability

Chapter.5- Conclusion and further work