Task offloading in Mobile Edge Clouding networks
Introduction

• Emerging compute-intensive and latency-sensitive applications, such as VR

• Mobile Edge Computing (MEC) extends computation resources to the edge of mobile networks [1]

• Network Functions Virtualization (NFV) enables flexible network service delivery by creating demanded Virtualized Network Function (VNF) instances at cloudlets in MEC networks
Background

Figure 1: An example of a mobile edge cloud network
Motivations

• Share VNF instances among multiple users with same request service

• Admit more user requests while minimizing the maximum delay in the MEC network
Problem definition

Assuming each request need only one VNF:

- Problem 1: maximize the amount of admitted requests without considering the end-to-end delay, subject to the computing capacity constraint on each cloudlet in the MEC network.

- Problem 2: maximize the amount of admitted user requests while minimizing the maximum delay of request, considering the sharing of VNF instances, subject to the end-to-end delay requirements and computing capacity constraints in the MEC network.
Generalized assignment problem (GAP)

- Let $C=\{c_1, \ldots, c_i\}$ be the set of MEC cloudlets and $Q=\{q_1, \ldots, q_j\}$ be the set of tasks. Each cloudlet $c_i \in C$ has a resource capacity $\text{cap}_i$. For each $c_i \in C$ and each $q_j \in Q$, we are given resource requirements $r_{i,j}$ for assigning task $q_j$ to cloudlet $c_i$.

- The objective is to find an assignment which maximize the admitted requests, formulation as follows

$$\max \sum_{c_i \in C} \sum_{q_j \in Q} X_{i,j}$$

subject to

$$\sum_{q_j \in Q} r_{i,j} X_{i,j} \leq \text{cap}_i, \ \forall c_i \in C$$

$$\sum_{c_i \in C} X_{i,j} \leq 1, \ \forall q_j \in Q$$

$$X_{i,j} \in \{0, 1\}, \ \forall c_i \in C, \forall q_j \in Q$$
Offloading request end-to-end delay $D_i$

- Consist of: network transmission delay of shortest path $t_{i,j}$, VNF processing delay $p_{i,j}$, and instantiation delay for creating new VNF instance $o_i$ while admitting request $q_i$ in the cloudlet $c_j$

- Share existing VNF instance,

\[ D_i = p_{i,j} + t_{i,j} \]

- Create new VNF instance,

\[ D_i = p_{i,j} + t_{i,j} + o_i \]
Min-Cost Max Flow

• Set cost as the delay time $D_i$ of admit request $q_i$ in the cloudlet $c_j$
Method

- Use ILP formulation to get exact solution
- Devise efficient heuristic algorithm for each case
Reference