# Mobile Edge Computing in Massive IoT Systems

Muhammad Asif Khan[1], Emna Baccour[2], Ridha Hamila[3], Aiman Erbad[2]
Qatar Mobility Innovations Center[1], Hamad Bin Khalifa University[2], Qatar University[3]

IEEE International Symposium on Networks, Computers and Communication

October 23 - Doha, Qatar

# Topics

- Introduction
- Architecture
- Deployments and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

# Topics

- Introduction
- Architecture
- Deployments and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

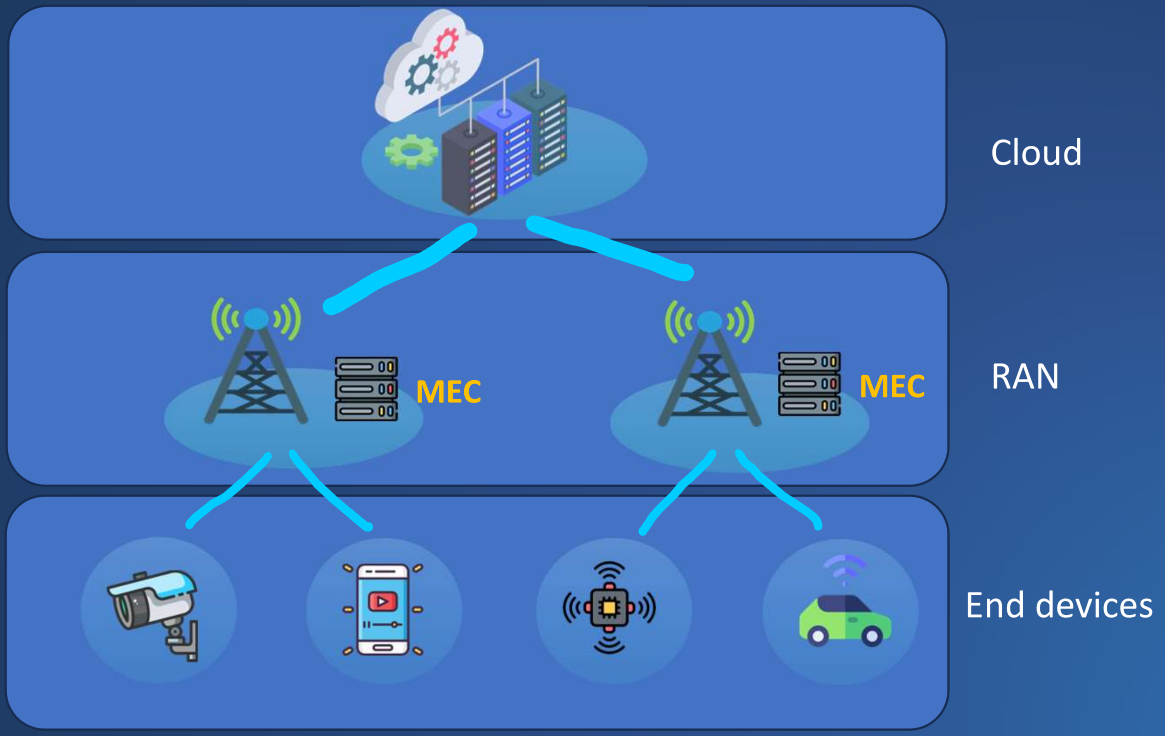
# Mobile Edge Computing

"A computing paradigm that provides information technology (IT) and cloud computing capabilities at the edge of the mobile network, within the Radio Access Network (RAN) in close proximity to mobile subscribers"

*ETSI ISG MEC*

MEC is now called ~~Mobile Edge Computing~~ "**Multi-access Edge Computing**".

# Mobile Edge Computing



# Cloud Computing - Challenges

- Latency
  - Higher data transmission and queuing delays.
  - **Example:** video frames from an autonomous vehicle need to be processed in milliseconds t avoid obstacles e.g., 200 ms to send and process a camera frame at AWS server.
- Scalability
  - Bulk data transmission to the cloud can create network bottlenecks.
  - All data might not be needed at the cloud (e.g., deep learning training).
- Privacy
  - Data might be sensitive to the user (e.g., speech, images, documents).
  - User does not know if data is being illegally used by the cloud provider?

# MEC Benefits

Latency
Scalability
Privacy

- ➢ Mobile Edge computing can solve these issues.
  - ✓ Lower transmission and queuing delay minimizes latency.
  - ✓ Edge resources can be managed as per demand.
  - ✓ Data being local with MNO can be regulated.

**Exclusive MEC Benefits**

- ✓ Real-time access to radio network information
- ✓ Location aware

# Related Concepts

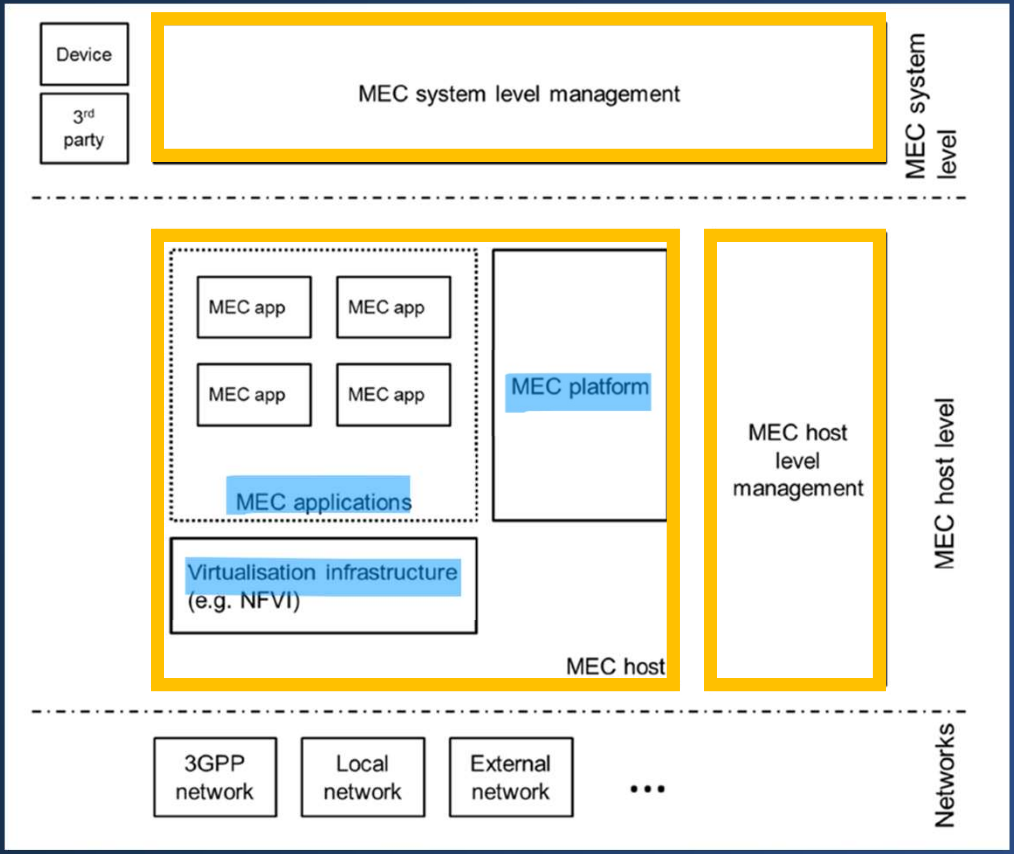
- Fog Computing
  - The term was coined by Cisco and was mainly used in the context of IoT.
  - IoT gateway is the fog node.
  - Less frequently used than MEC.
- Cloudlet
  - Placing computing resources farther than the edge and closer than the cloud.
  - Some literature has a different definition for cloudlet.

- The term MEC is more popular, standardized (3GPP) and widely used.

# Topics

- Introduction
- Architecture
- Deployment and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

# MEC ETSI Architecture



### 1. MEC Host contains:

- **Applications** - VMs running on top of NFV infrastructure.
- **NFVI** – provides compute, storage, and network resources for apps.
- **Platform** - an environment where MEC applications

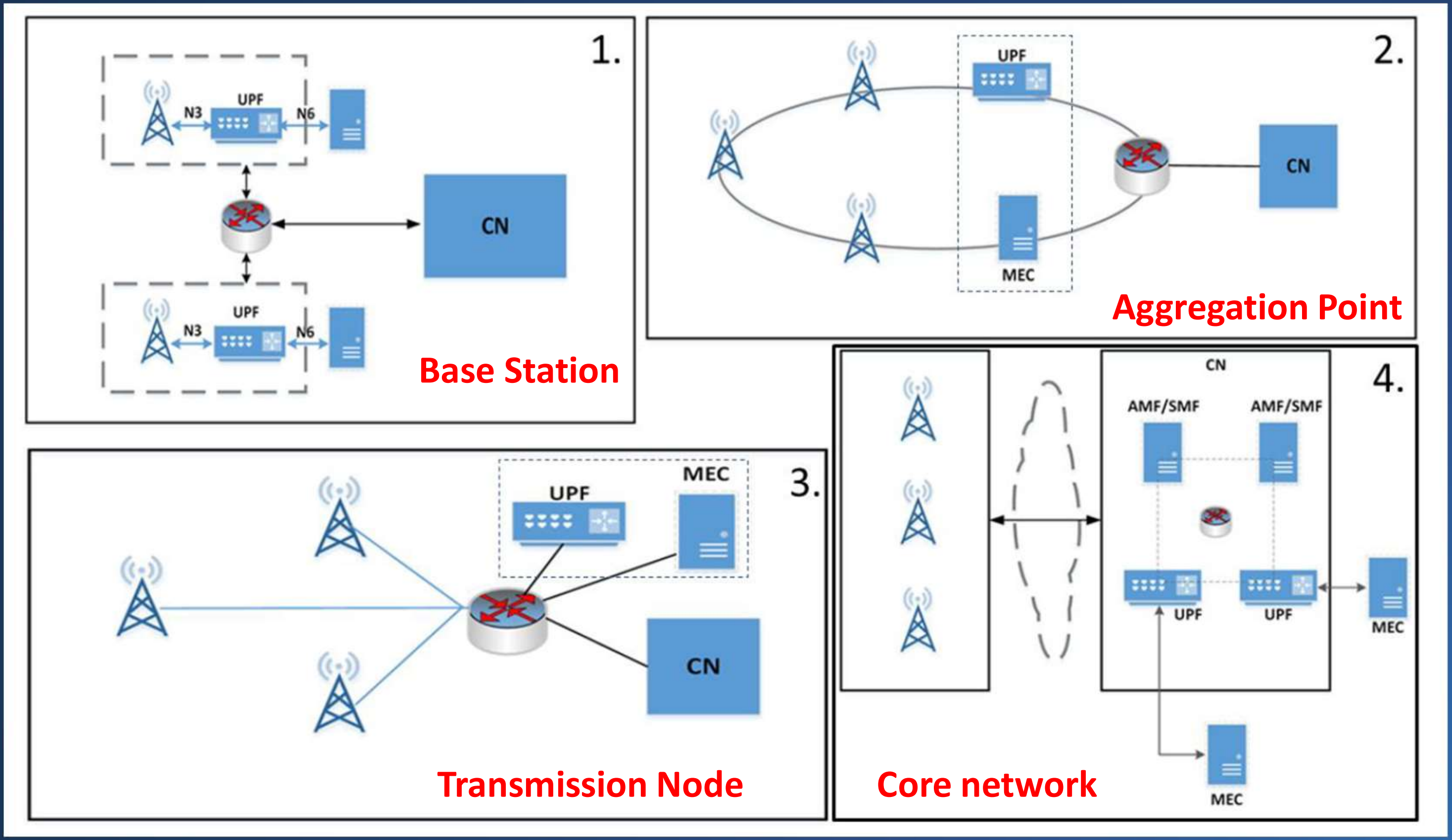
### 2. MEC Host level management

- Traffic routing among apps, services, and networks

### 3. MEC System Level Management

- Orchestrator for maintaining MEC hosts, resources management, service management, etc.

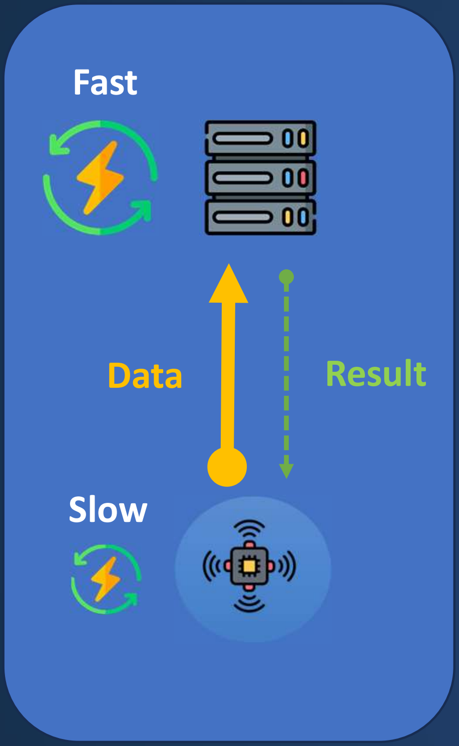
# MEC Architecture

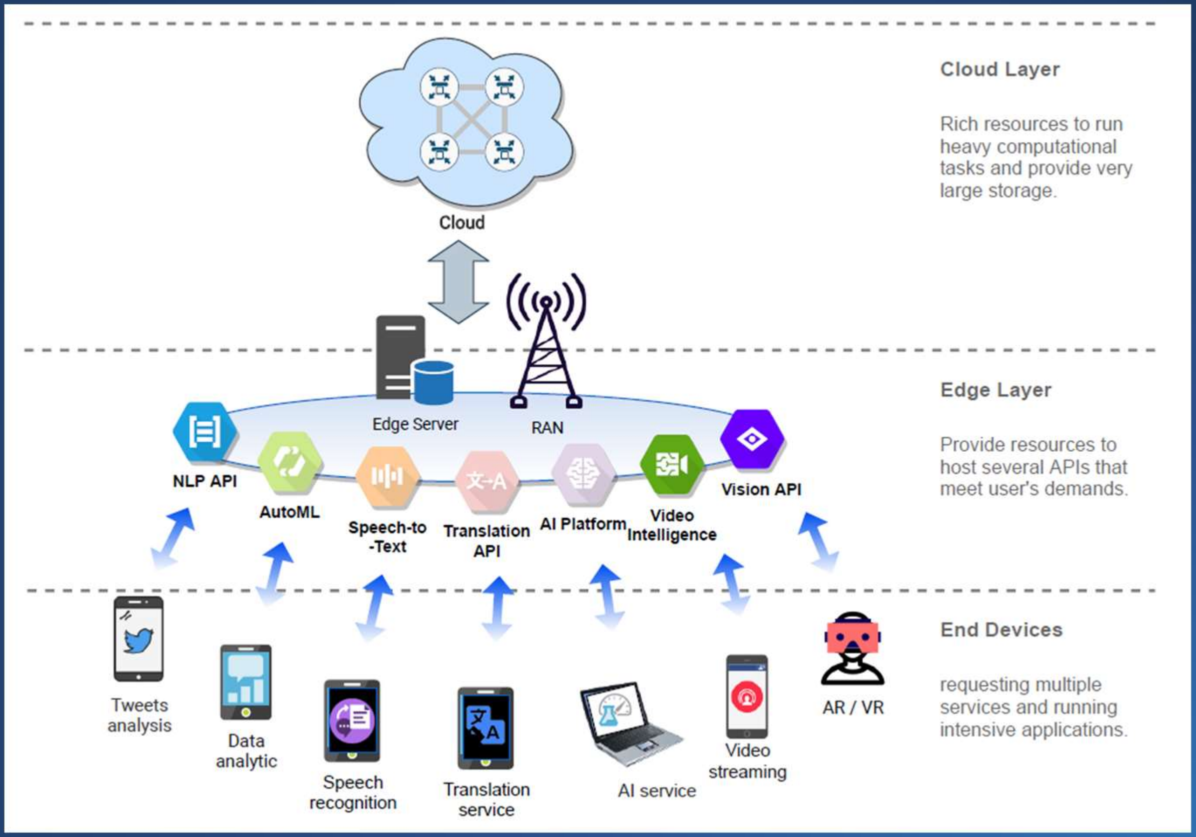


# Topics

- Introduction
- Architecture
- Deployments and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

# MEC Use Cases





**Cloud Layer**

Rich resources to run heavy computational tasks and provide very large storage.

**Edge Layer**

Provide resources to host several APIs that meet user's demands.

**End Devices**

requesting multiple services and running intensive applications.

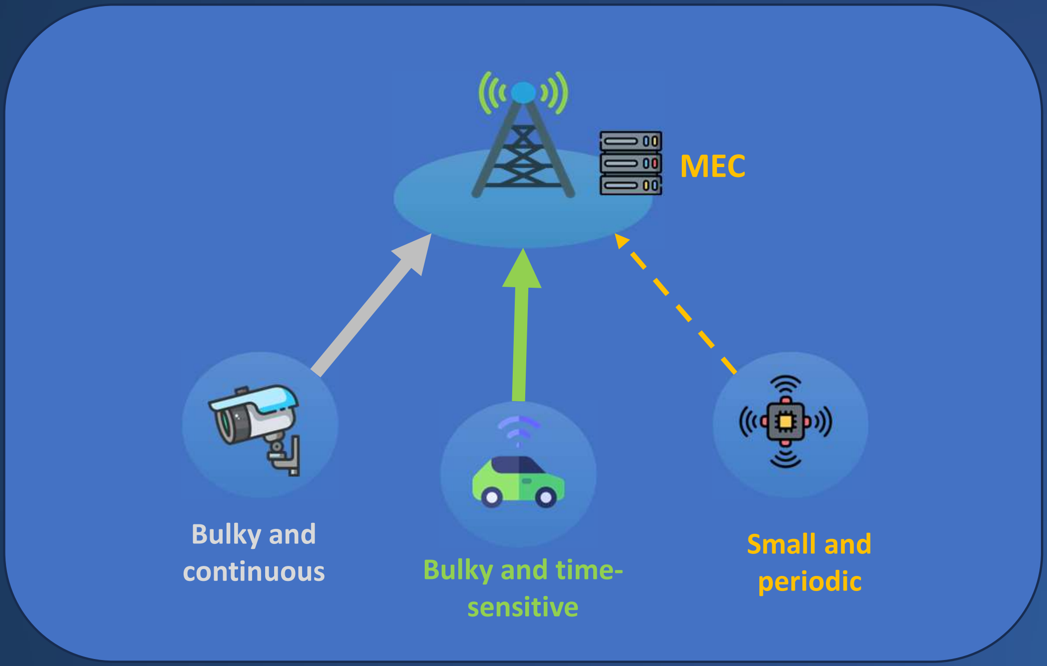
# Some examples

- Location Tracking
  - Location-based services for retail locations with no GPS coverage.
- Massive IoT
  - MEC servers across a geographical area (campus, city, network-wide) receive huge amounts of raw data from sensors and devices, process it, extract semantic information, send the information to the central cloud, and may cache for a certain period of time. For examples,
  - Surveillance and monitoring during events (face recognition data, abandoned baggage, cars data)
- Application Computation Offloading
  - MEC host executes the computation-intensive functions on behalf of mobile devices
  - Graphical rendering in AR/VR, 3D gaming, and video analytics

# Topics

- Introduction
- Architecture
- Deployments and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

# Network Slicing



# Network Slicing

A network slice is a logical network that provides specific network capabilities and network characteristics. (3GPP TS 23.501).”

- A “**5G Slice**” can span all domains of the network, software modules on the cloud, transport network, RAT settings, and configuration of the 5G device.

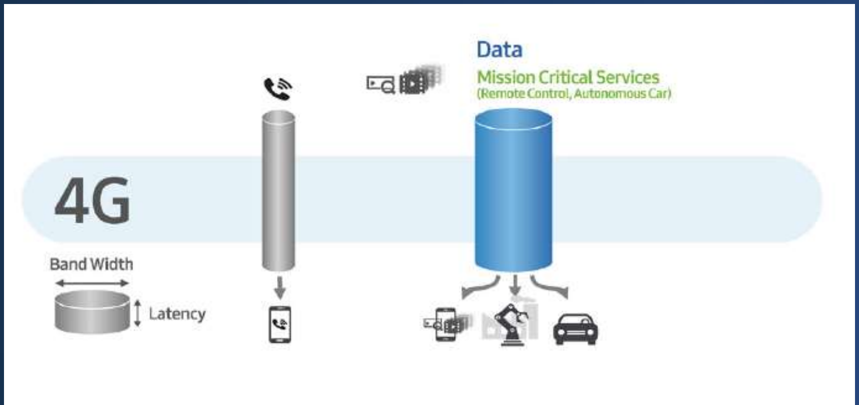
# Network Slicing - Introduction

- Allows CSPs to monetize services by creating discrete network slices that meet required levels of performance for individual customers and services.

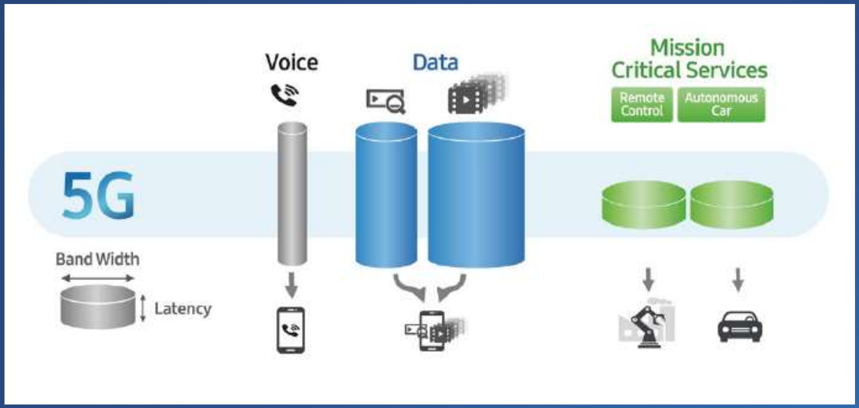
- **Customized networks:**
  - Customized functionalities - priority, charging, policy control, security, and mobility.
  - Customized performance - latency, mobility, availability, reliability and data rates.
  - Restricted access - MPS users, public safety users, corporate customers, roamers, or hosting an MVNO.

# **Network Slicing** - Introduction

A single pipeline for data services (e.g., video streaming, Internet surfing and navigation).

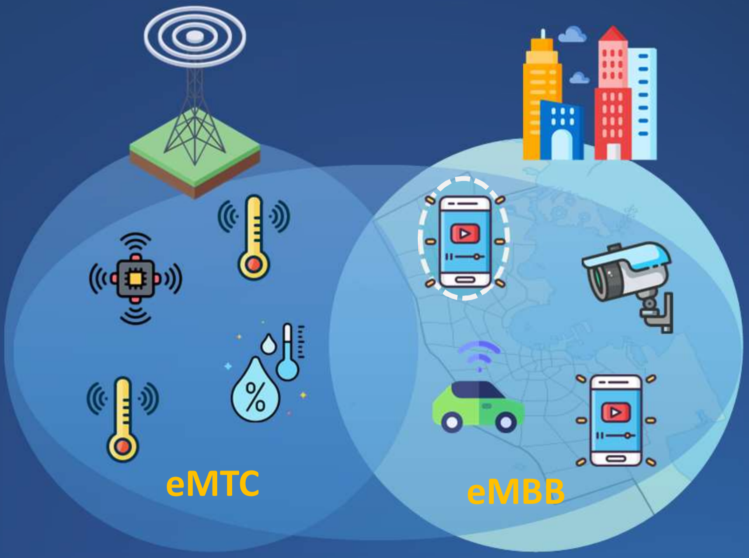


Virtual data pipelines for each data service.



# Network Slicing - Requirements in 5G

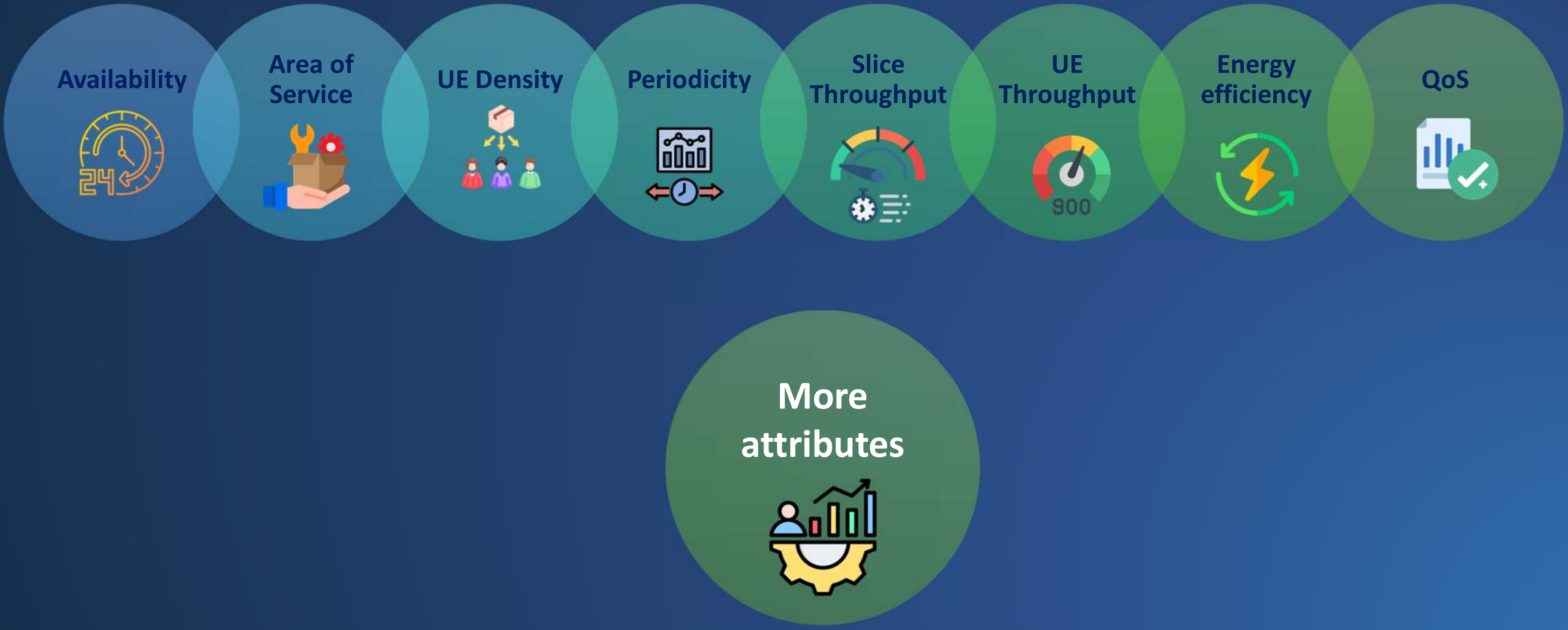
- MNOs can
  - Create/modify/delete slices
  - Define/update new services.
  - Add/move/remove users to one or multiple slices.
  - Scale NS without impacting the “minimum available capacity” of other network slices.
  - Define service priorities for slices.
  - Restrict geographical boundaries for a slice.
  - Limit a user to only receiving service from an authorized slice.
  - Creation, modification, and deletion of a network slice shall have no or minimal impact on traffic and services in other network slices in the same network.



Medium priority

High priority

# Network Slicing - Attributes



Availability

Area of Service

UE Density

Periodicity

Slice Throughput

UE Throughput

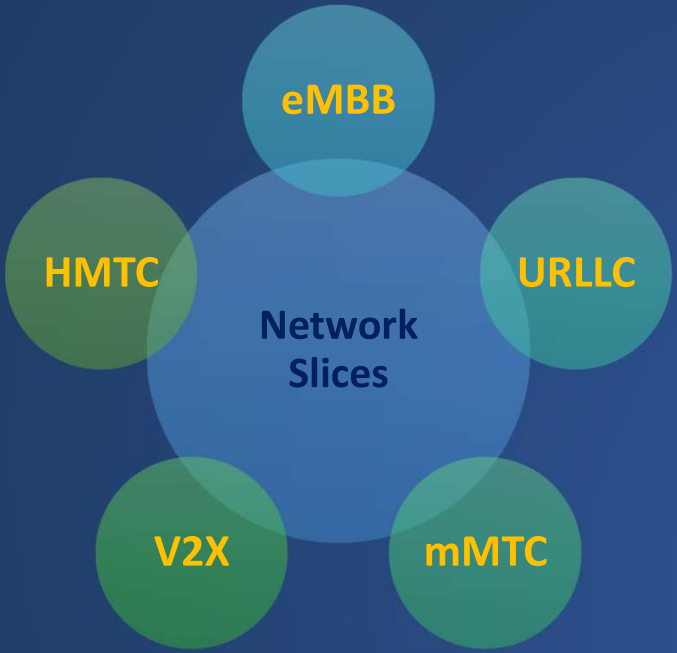
Energy efficiency

QoS

More attributes

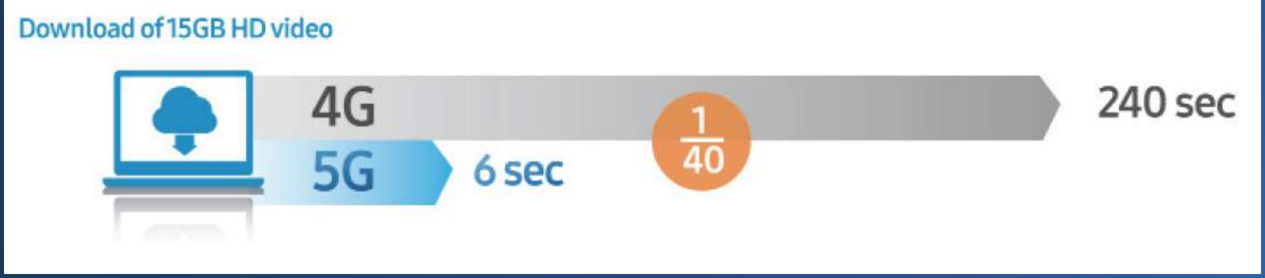
# Network Slicing - Types

- 3GPP defines **Five Types** of network slices



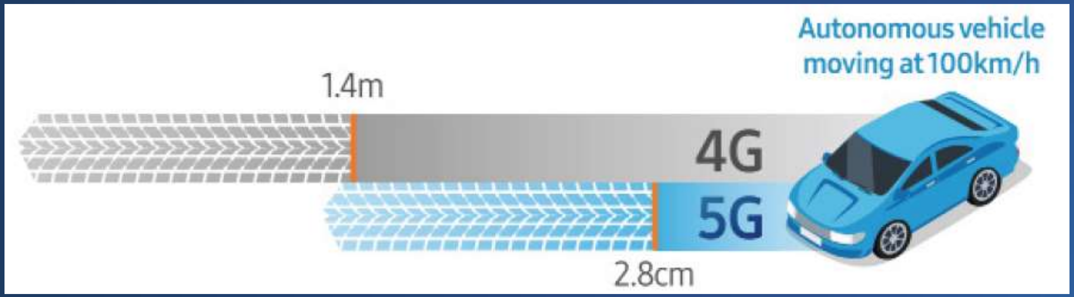
# Network Slicing - eMBB

- **enhanced Mobile BroadBand**
- Typical data rates from 100Mbps up to 20Gbps per user
- 100Mbps data speed at the cell edge (where user receive a weak signal).
- **Applications**: High definition (HD) videos, virtual reality (VR), and augmented reality (AR).



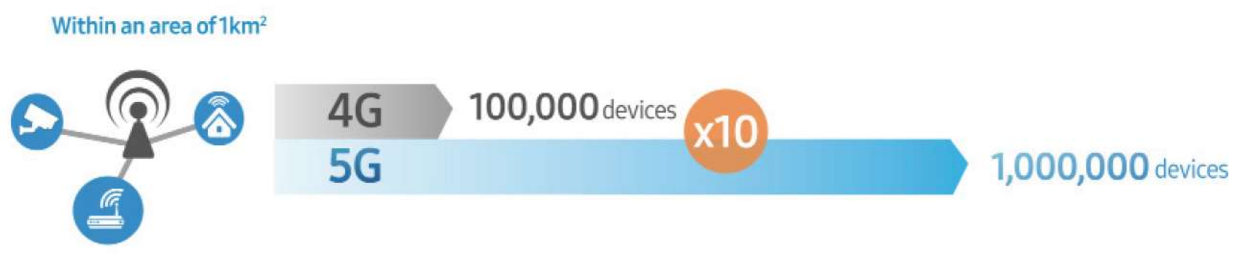
# Network Slicing - URLCC

- **Ultra Reliable and Low Latency Communications**
- 1 millisecond latency (10ms in 4G)
- 99.99% reliability
- **Applications**: Remote robot control, smart health, autonomous vehicles, etc.



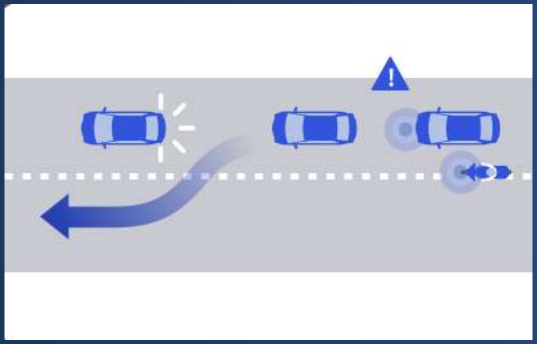
# Network Slicing - mMTC

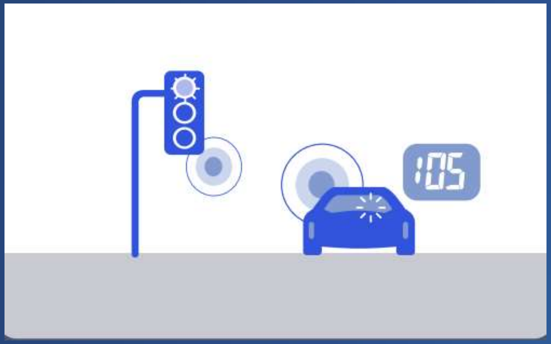
- **massive Machine Type Communications**
- Small packets, low-rate, uplink-centric transmission, tolerate high latency (~10s)
- 1 million IoT devices per $Km^2$ (10X more than 4G).
- **Applications**: smart buildings, smart HVAC, smart lighting, environmental monitoring, fire detection.



# Network Slicing - V2X

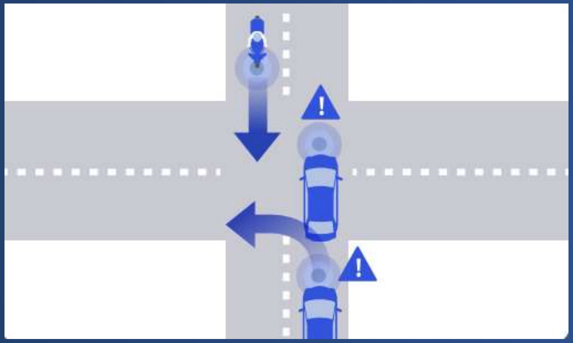
- **Vehicle-to-everything** (e.g., V2V , V2I, V2P, V2N)
- A customized slice for V2X services
- Distance-based multicast communication
- **Applications**: cooperative traffic management, electronic toll system, road safety, UAV communication





# Network Slicing - HMTC

- **High performance Machine-Type Communication**
- Mixed requirements that do not fit into any of the particular slices e.g., large data (~eMBB), low latency (~URLLC), high density (~mMTC)
- delay <10ms, fixed position devices, density < 1000/km2, mission-critical support
- **Applications**: industrial automation, public safety, remote robotic surgery



# Network Slicing meets MEC

- MEC + Network Slicing → A natural integration
- MEC assures the end-to-end network latency for a network slice instance (A transport network can not guarantee latency).
- Dedicated MEC host for a network slice for further assurance of KPIs.
- Multi tenancy

# Topics

- Introduction
- Architecture
- Deployments and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

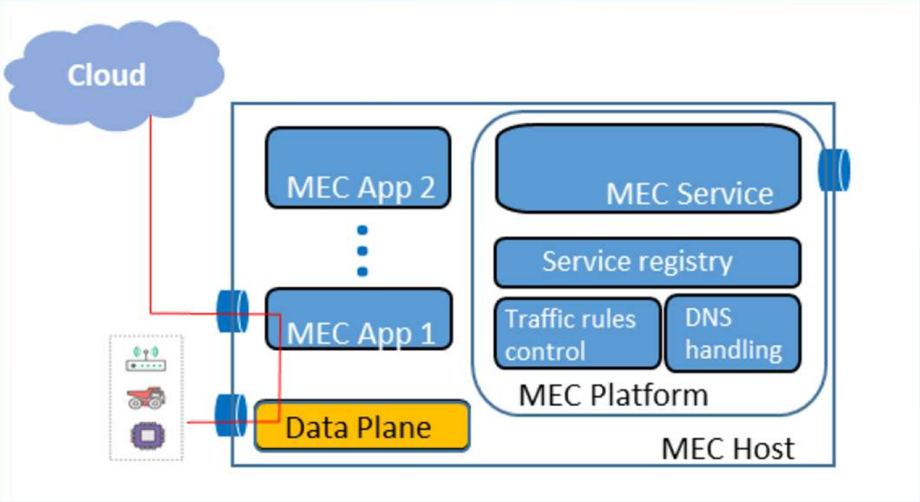
# Massive Internet of Things (MIoT)

- ~ 83 billion IoT devices by 2024 (Juniper). Some estimated 64 billion.
- MIoT is ~ mMTC in the 5G world.
- Critical IoT ~ URLLC in the 5G world.

- We keep the discussion open to large-scale (~massive) IoT systems.

# MEC in Massive IoT

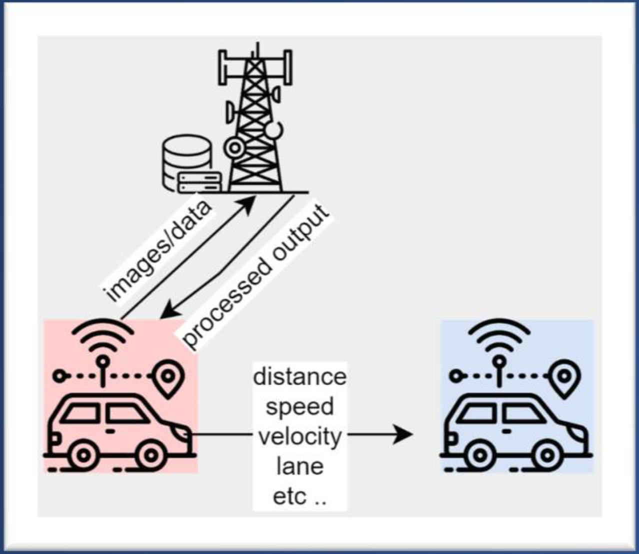
- MEC enables serverless computing for MIoT devices by hosting Function as a Service (FaaS) in the edge.
- Further support integration with the Cloud.



MEC in 5G Networks, ETSI White Paper No. 28

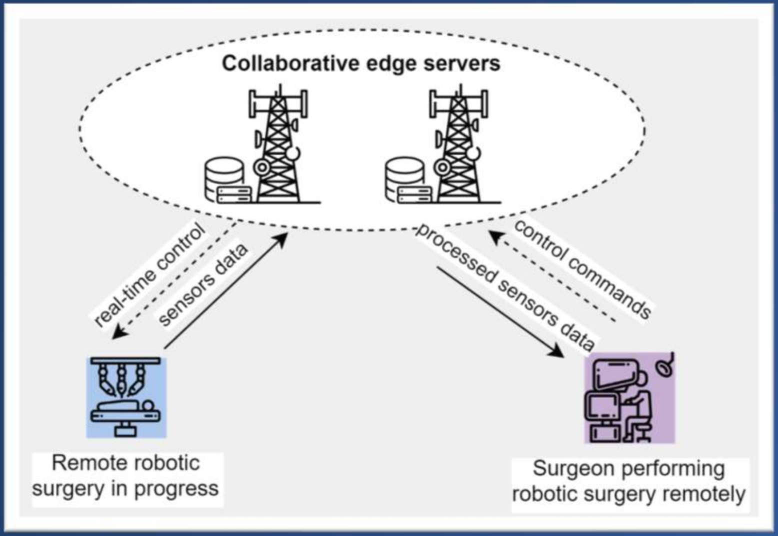
# MEC in MIoT – Use Cases

## Vehicle to Infrastructure (V2X)



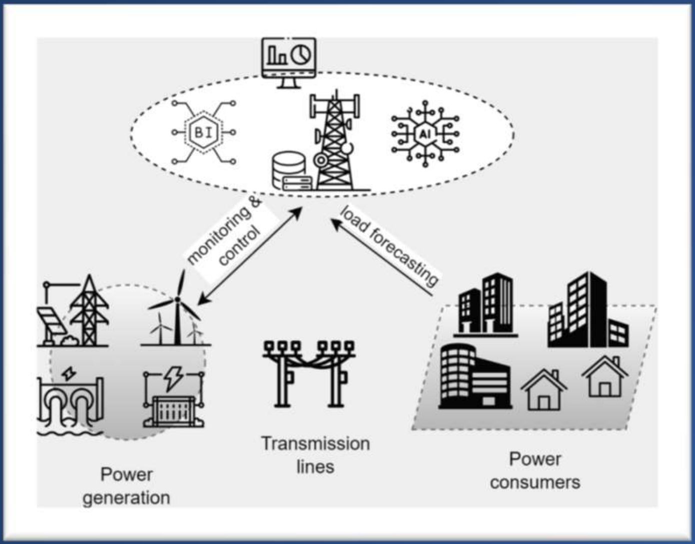
# MEC in MIoT – Use Cases

## Healthcare


Collaborative edge servers
real-time control
sensors data
processed sensors data
control commands
Remote robotic surgery in progress
Surgeon performing robotic surgery remotely

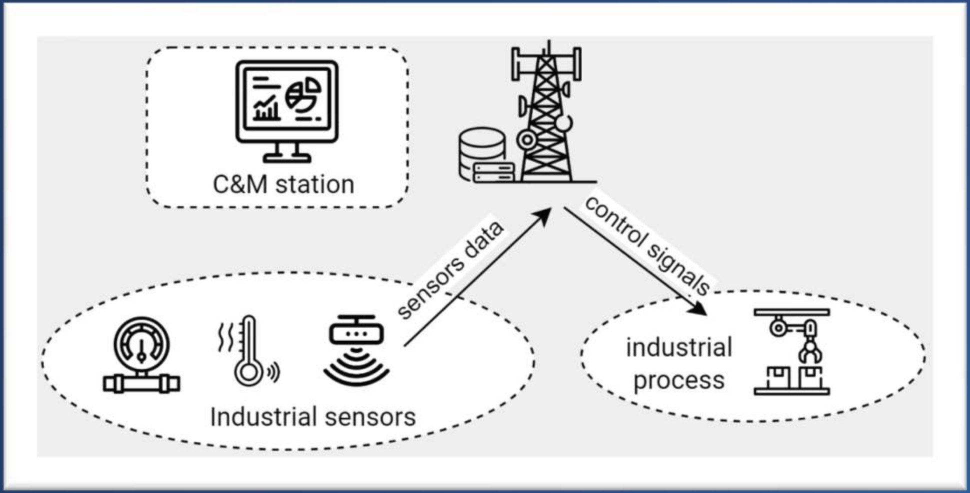
# MEC in MIoT – Use Cases

## Smart Grids



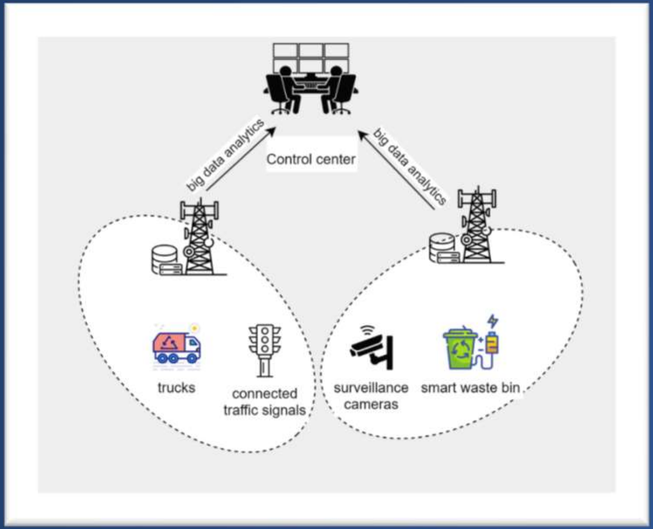
# MEC in MIoT – Use Cases

## Manufacturing

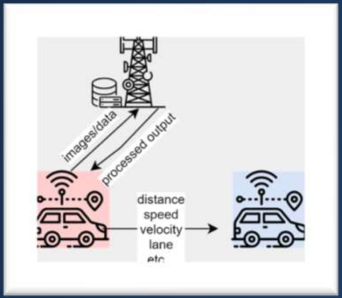


# MEC in MIoT – Use Cases

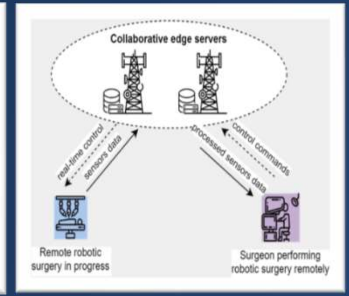
## Smart Cities



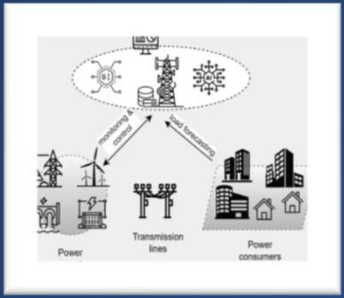
# MEC in MIoT - Use Cases



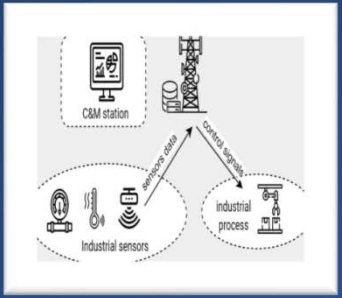
V2X


Collaborative edge servers
real-time control
sensors data
processed sensors data
control commands
Remote robotic surgery in progress
Surgeon performing robotic surgery remotely

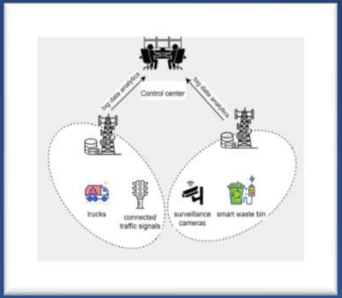
Healthcare



Smart grids



Manufacturing



Smart cities

**Evolution of MEC Systems …**

Khan et al., "**"Mobile Edge Computing for the Next Generation Massive Internet of Things**", in EICCS 2023 (Book Chapter)

# Topics

- Introduction
- Architecture
- Deployments and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

# MEC System

## Two primary functions of MEC

1. Data offloading (content caching)
2. Computation offloading

Khan et al., "A survey on mobile edge computing for video streaming: Opportunities and challenges", in IEEE Access 2022

# Content Caching

- Traditional Caching Approaches (Content caching)
  - Most Popular Videos (MPV) – based on nationwide popularity
  - Least Recently Used (LRU) – Replaces the Idle video with new videos
  - Least Frequently Used (LFU) – no. of requests to measure popularity over a timeframe
  - Least Recently Frequently Used (LRFU) – Frequency and recentness of a video
  - First In First Out (FIFO) – Caching according to the order of the arrival of videos
- Behavior-aware Caching
  - Proactive popularity – Probability of future popularity and requests
  - Social dynamics – Based on factors such as likes, shares, and friend circles to find trends.
- Cooperative Caching
  - Servers cooperate to serve network-wide user requests.

# Content Caching - PcP

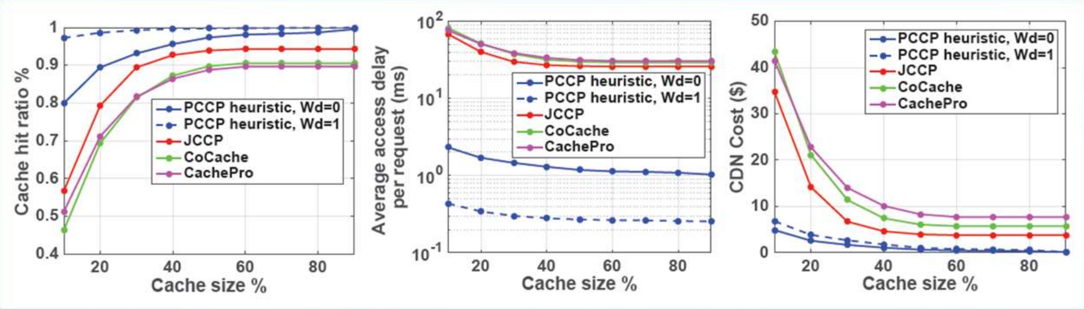
- Users would not watch the whole video but rather watch the first few chunks.

- Proactive Cache Policy (PcP)
  - Popularity-aware proactive caching
  - Chunk popularity (user preferences) instead of the whole video popularity.

Emna et al., "Proactive video chunks caching and processing for latency and cost minimization in edge networks", in IEEE WCNC 2019

# Content Caching - PCCP

- Proactive Caching and Chunk Processing (PCCP)
  - Joint video caching and transcoding
  - Proactively fetch video chunks based on chunk popularity.
  - Chunks shared stored on neighboring edge servers via X2 interface.
  - Upon request, chunks corresponding to the same video are collected from the neighboring servers and served to the user.



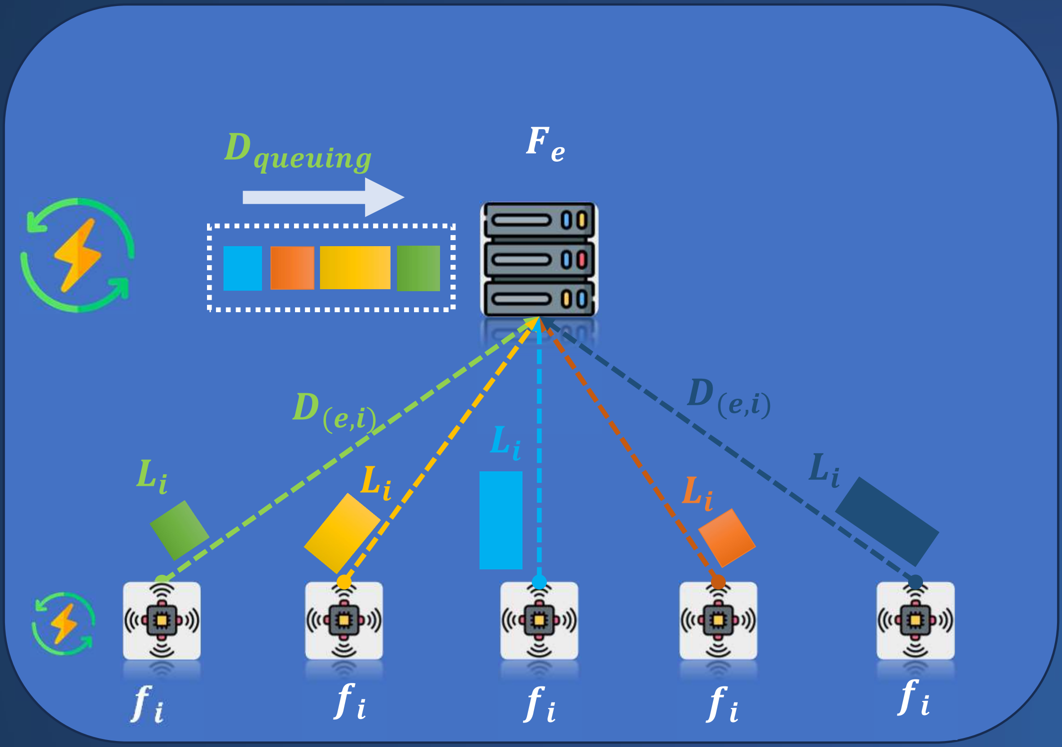
Reduce backhaul link usage, delay, and CDN cost.

Kashif et al., "Collaborative joint caching and transcoding in mobile edge networks", in JNCA 2019

# Computation Offloading

- Single-server systems
  - Binary vs Partial offloading – Offload full task or portion of task
  - Deterministic vs Stochastic offloading – Predetermined or randomly taken decision
  - Server scheduling – Which task is to be done first?
- Multi-server systems
  - Server selection – Which server to send requests?
  - Server cooperation – Servers cooperate to serve a pool of users connected to different RANs.
  - Server migration – Transfer ongoing tasks or associated users to another server.

# Computation Offloading – System Model



# Computation Offloading

- Local computing: Time to process a task of size $L$ locally at a user device $i$ with processing power $f_i$ ($C$ is the no. of CPU cycles required to process a single bit)

$$D_i^k = \frac{L_i^k C}{F_i}$$

- Edge computing: Time to process the task of user $i$ which is of size $L$ at the edge server with processing power $F_i$ ($C$ is the no. of CPU cycles required to process a single bit)

$$D_e^k = \frac{L_i^k C}{F_e}$$

# Computation Offloading

- Communication Delay: The transmission delay to transmit the $k^{th}$ task from the user device to the edge server.

$$D_{(e,i)}^{k} = \frac{1}{R_{(e,i)}^{k}}$$

$$R_{(e,i)}^{k} = W \log_2(1 + \frac{P_e \left|h_{(e,i)}\right|^2}{N_i})$$

- There is also "queuing delay" when there are more than one users sending requests to the same server simulataneously.

# Computation Offloading

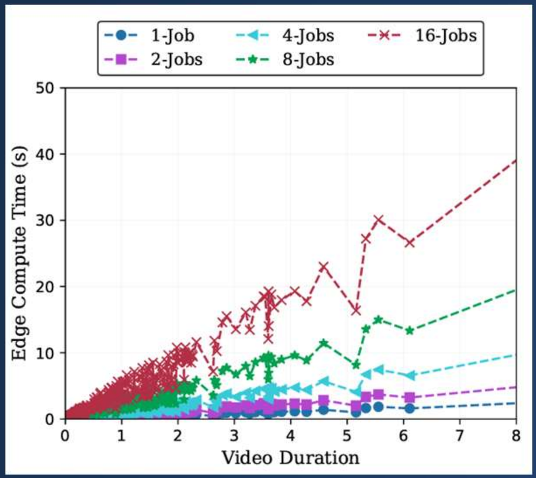
- Total Edge Computing Delay:

$$D_{edge} = D_{transmission} + D_{queuing} + D_{processing}$$

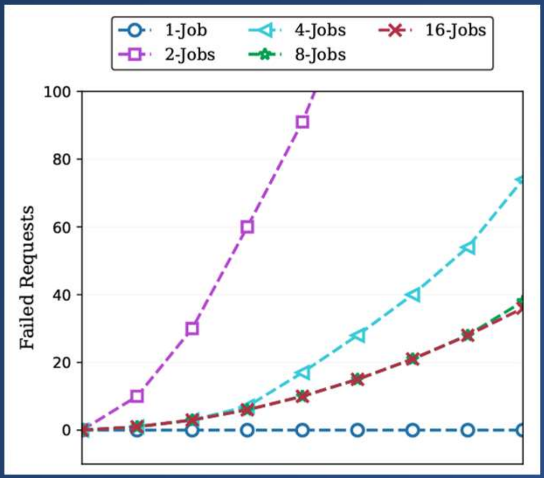
- Offloading Decision:

$$\begin{cases} D_{edge} \leq D_{local} & \text{Offload} \\ D_{edge} \geq D_{local} & \text{Do not offload} \end{cases}$$

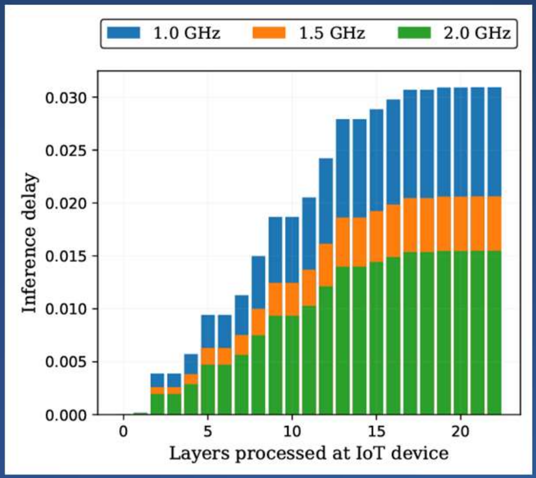
# Computation Offloading - Evaluation



Video transcoding delay at increasing simultaneous tasks.



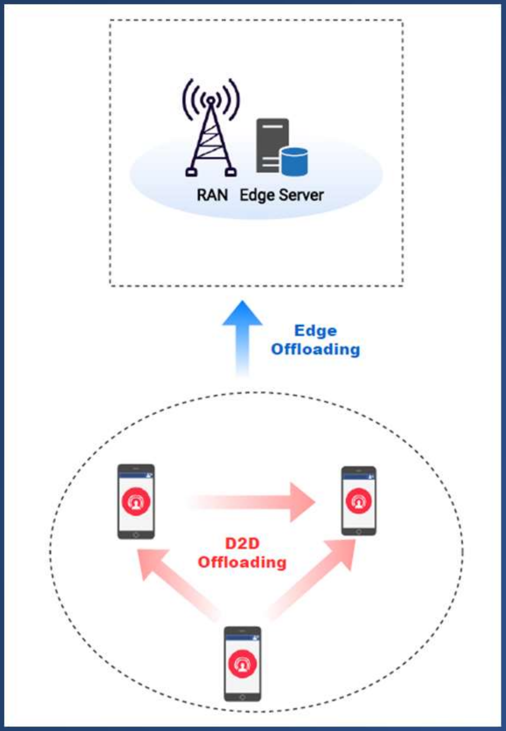
No. of failed requests (delay deadline did not meet).



Inference delay using distributed computing.

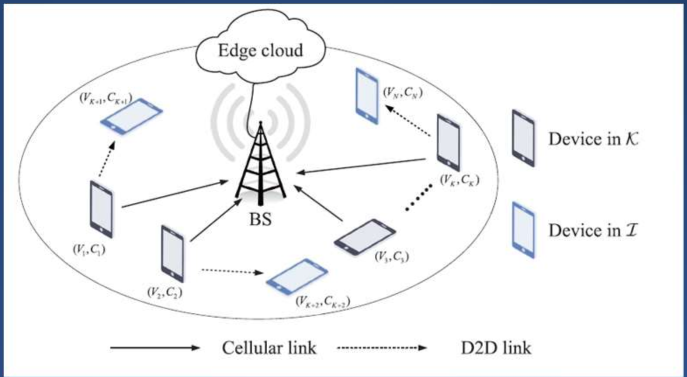
Mariam et al., "Edge Computing in IoT: A 6G Perspective", in Arxiv.org

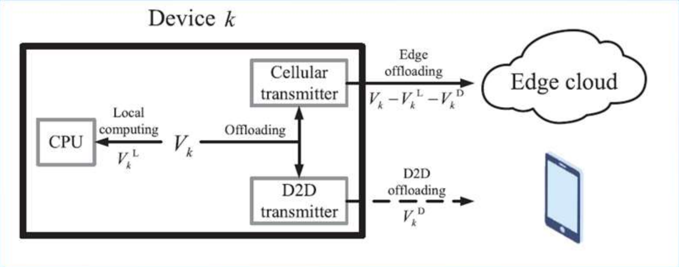
# D2D-MEC Systems



# D2D-MEC Systems

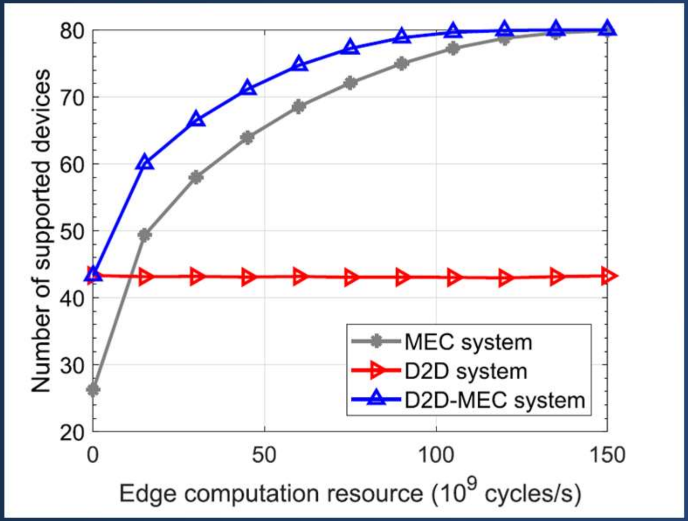
- D2D can improve the capacity of MEC systems.
- A device can offload a task to the edge or a nearby device using D2D.
- **Maximize** the number of supported devices:
  - communication constraint
  - computation constraint.
- Problem formulation:
  - mixed integer non-linear problem (MINLP)
  - Split into two sub-problems



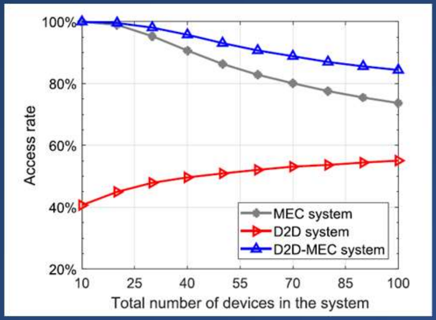


He et al., "D2D Communications Meet Mobile Edge Computing for Enhanced Computation Capacity in Cellular Networks", in IEEE Transactions on Wireless Communication 2019

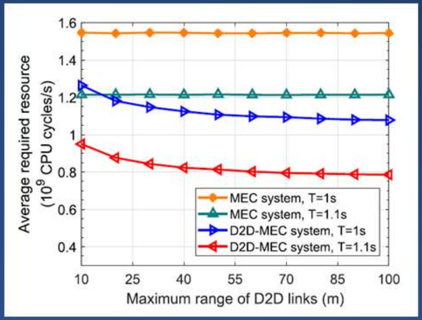
# D2D-MEC Systems



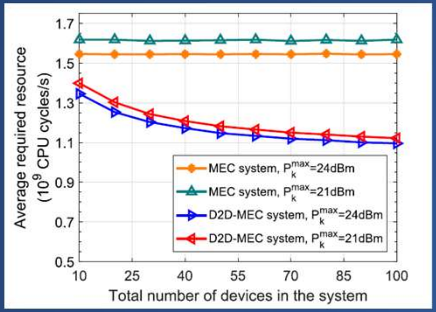
D2D-MEC supports more devices than MEC system.



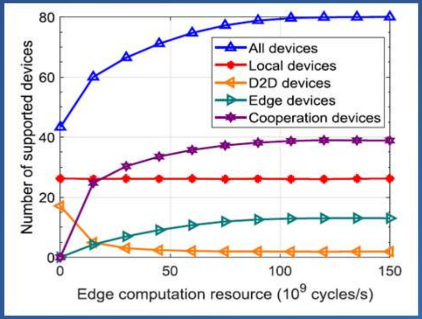
Higher access rate for D2D-MEC



More D2D-links, less edge resources



D2D-MEC requires less edge resources



More edge nodes, more devices served.

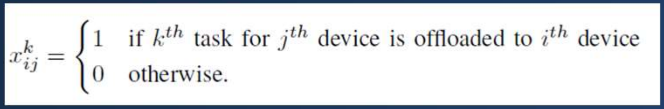
# Computation Offloading in D2D Edge (CODE)

- Thundering-herd problem
  - Unplanned sudden spikes in computation tasks
  - Edge servers are utilized up to the full capacity
- Edge can offload computations to users with capacity.
- Two algorithms
  - MOD – Maximum Offloading with Delay Constraint
    - Guarantee delay threshold and maximize offloading.
    - suitable for delay-sensitive applications e.g., Live streaming
  - MDO – Minimum Delay Constraint
    - Relax delay constraint but achieves higher offloading than MOD.
    - Suitable for VoD streaming

A 45 min Facebook Live video reached ~ 800,000 live viewers

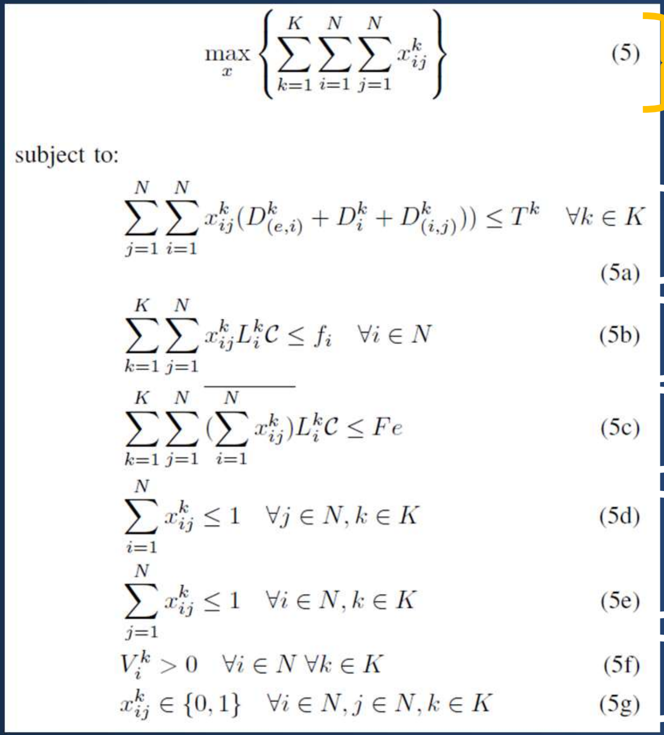
Khan et al., "CODE: Computation Offloading in D2D Edge System for Video Streaming", in IEEE Systems 2022

# MOD algorithm



$$x_{ij}^k = \begin{cases} 1 & \text{if } k^{th} \text{ task for } j^{th} \text{ device is offloaded to } i^{th} \text{ device} \\ 0 & \text{otherwise.} \end{cases}$$

Decision variable



$$\max_{x} \left\{ \sum_{k=1}^{K} \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij}^k \right\} \tag{5}$$

subject to:

$$\sum_{j=1}^{N} \sum_{i=1}^{N} x_{ij}^k (D_{(e,i)}^k + D_i^k + D_{(i,j)}^k)) \leq T^k \quad \forall k \in K \tag{5a}$$

$$\sum_{k=1}^{K} \sum_{j=1}^{N} x_{ij}^k L_i^k \mathcal{C} \leq f_i \quad \forall i \in N \tag{5b}$$

$$\sum_{k=1}^{K} \sum_{j=1}^{N} (\overline{\sum_{i=1}^{N} x_{ij}^k}) L_i^k \mathcal{C} \leq Fe \tag{5c}$$

$$\sum_{i=1}^{N} x_{ij}^k \leq 1 \quad \forall j \in N, k \in K \tag{5d}$$

$$\sum_{j=1}^{N} x_{ij}^k \leq 1 \quad \forall i \in N, k \in K \tag{5e}$$

$$V_i^k > 0 \quad \forall i \in N \; \forall k \in K \tag{5f}$$

$$x_{ij}^k \in \{0, 1\} \quad \forall i \in N, j \in N, k \in K \tag{5g}$$

Objective (maximize offloading)

Delay constraint (guarantee the delay deadline of the task)

Device capacity constraint

Edge capacity constraint

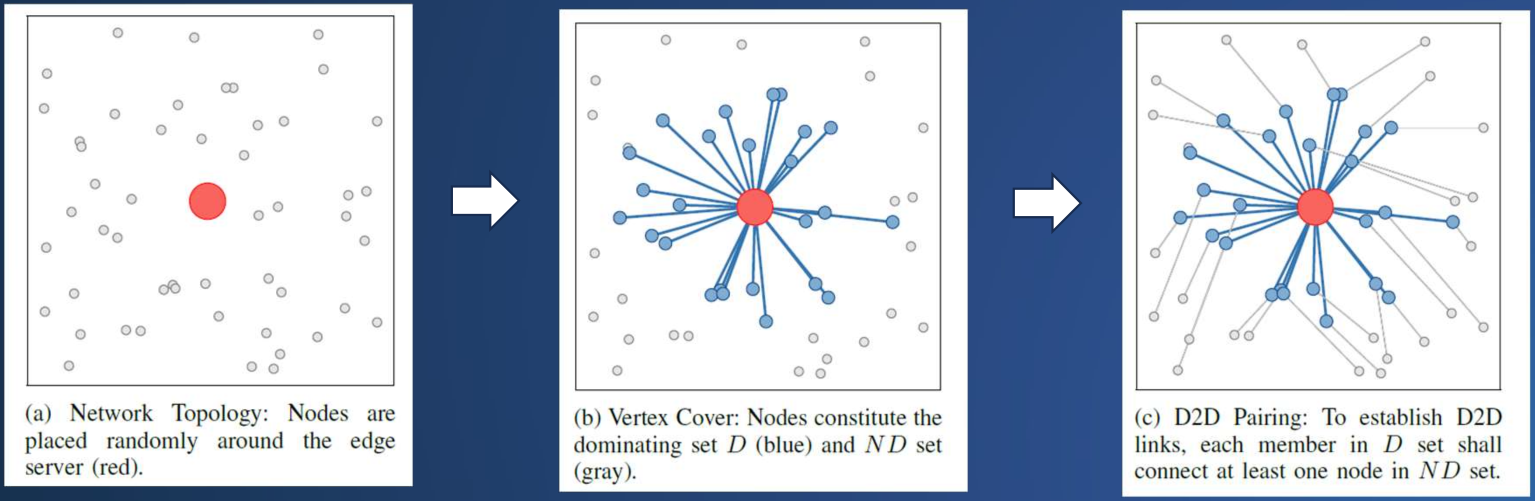
Assignment constraint

Variable limits

The problem is **NP-hard**.
Solved using a lightweight Heuristic.

# MDO algorithm

- Two stages:
  - Clustering: Minimum Weighted Dominating Set Problem (MWDSP)
  - D2D Pairing: Unbalanced Assignment Problem (UAP).



(a) Network Topology: Nodes are placed randomly around the edge server (red).

(b) Vertex Cover: Nodes constitute the dominating set $D$ (blue) and $ND$ set (gray).

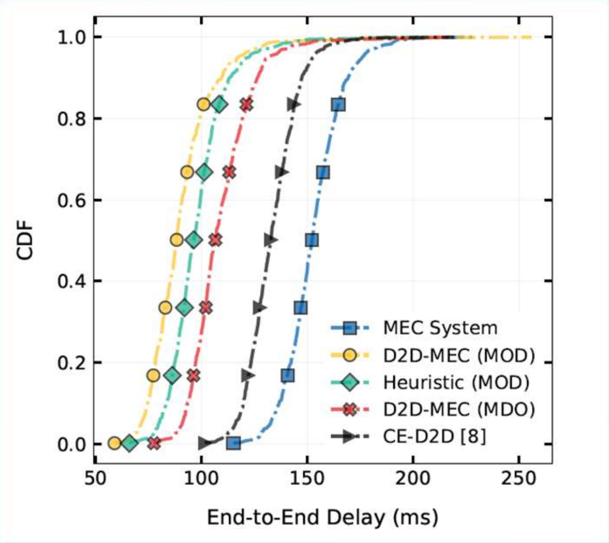
(c) D2D Pairing: To establish D2D links, each member in $D$ set shall connect at least one node in $ND$ set.

System of heterogeneous devices

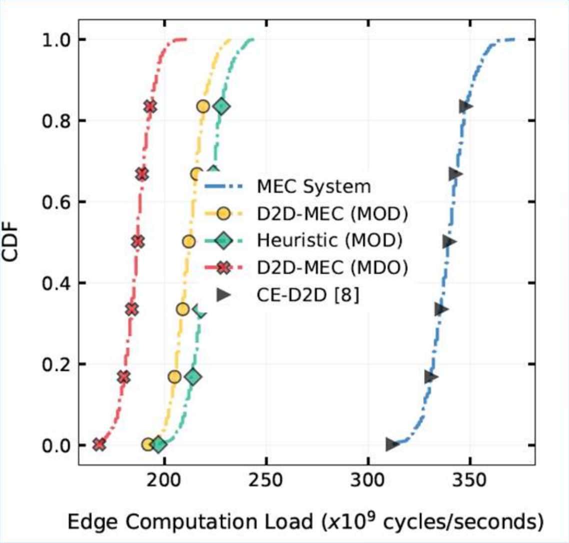
Dominating (D) set found.

D set connected to ND set.

# Computation Offloading in D2D Edge (CODE)

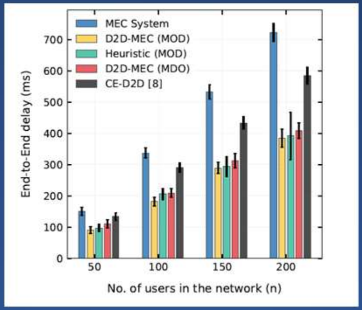


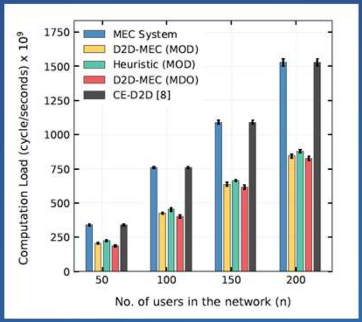
Delay: MOD outperforms MEC (65%) and MDO (22%)



Edge Load: MDO outperforms MEC and MOD

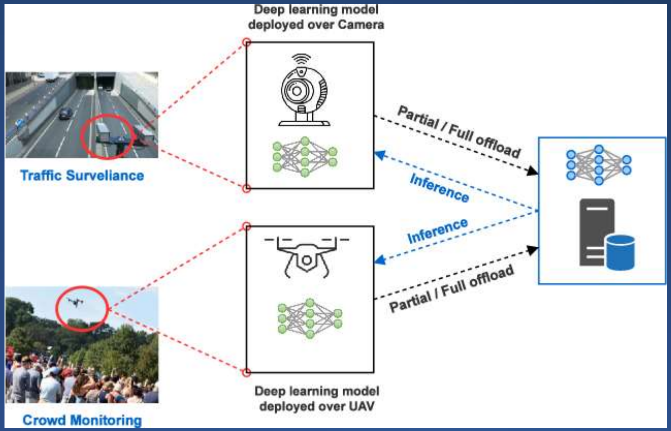
Higher gains for larger networks

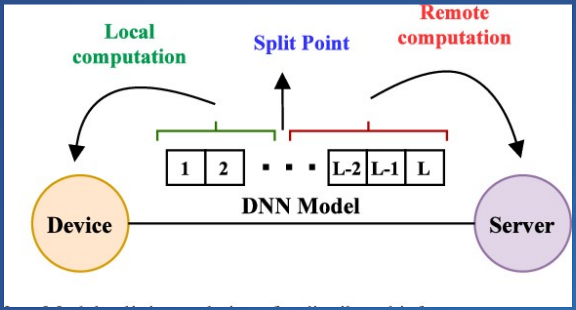




# Distributed Inference in Video IoT

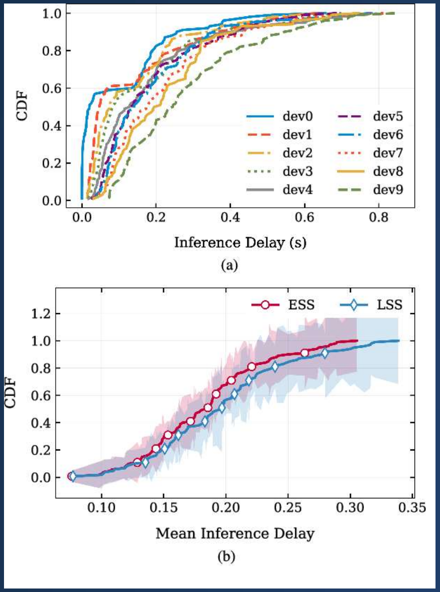
- IoT devices are resource-constrained – not suitable for running complex applications.
- Distributed inference over IoT devices and edge servers.
- Layer-based splitting of DNN.
- Two algorithms
  - Late Split Strategy (LSS): For IoT devices with a regular power source e.g., CCTV cameras (minimize delay).
  - Early Split Strategy (ESS): For battery-powered IoT devices e.g., drones (minimize energy).
- Evaluation:
  - Over VGG16 and MobileNet_V2 CNN models





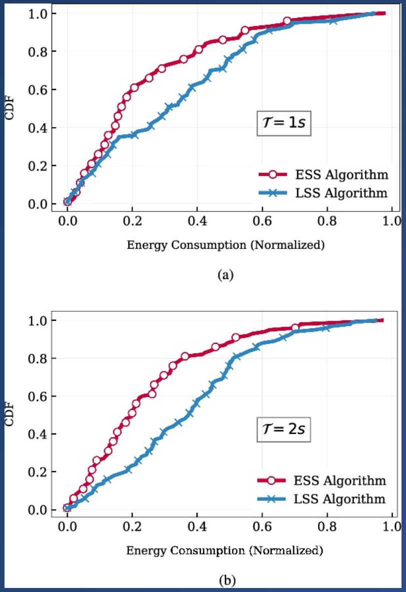
Khan et al., Distributed Inference in Resource-Constrained IoT for Real-Time Video Surveillance, IEEE Systems, 2022

# Distributed Inference in Video IoT



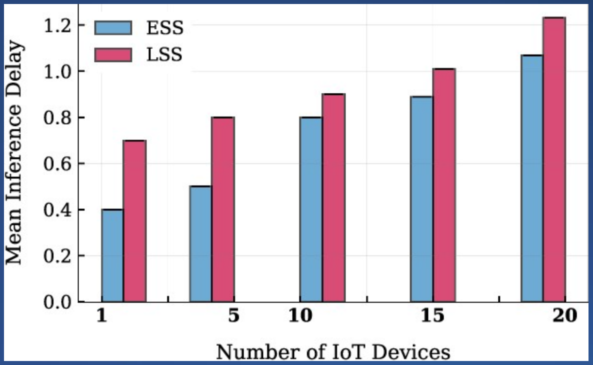
Inference Delay

Reduction ~20% (LSS) and ~60% (ESS).



Energy consumption

Reduction: ~18% (ESS) and ~52% (LSS).



Mean inference delay

Gain reduces when no. of devices increases.

# MEC with Network Slicing

- Minimize the aggregate completion time of computational tasks requested by WDs.
- Joint Slice Selection and Edge Resource Management (JSS-ERM)
  - Inter-slice radio allocation policy - AP resource shared by all slices.
  - Intra-slice radio allocation policy - Slice resources shared by WDs.
  - The problem is Mixed Integer Program ➔ NP-hard
  - Approximation ➔ Choose Offloading Slice (COS) algorithm.

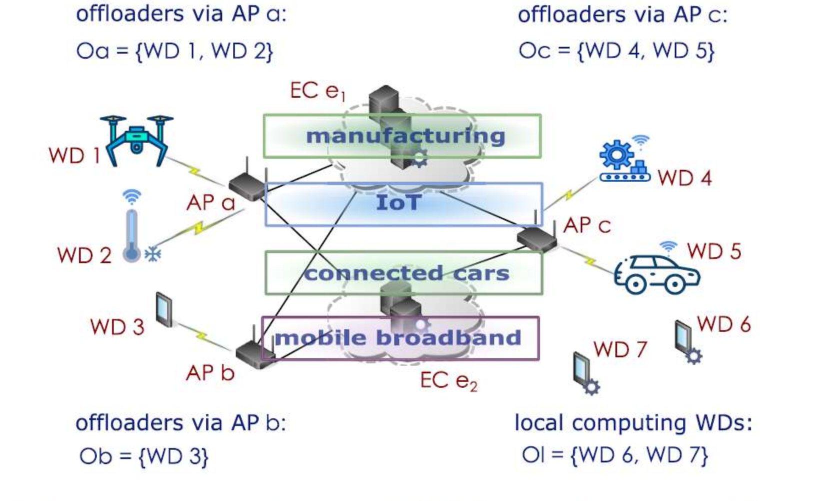
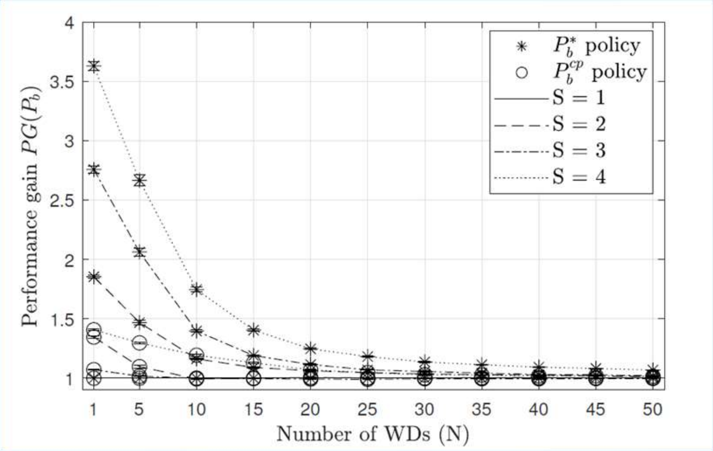


Fig. 1. An example of a slicing enabled MEC system that consists of $N = 7$ WDs, $C = 2$ ECs and $A = 3$ APs and $S = 4$ slices.

Sladana et al., Joint Wireless and Edge Computing Resource Management With Dynamic Network Slice Selection, in IEEE Transactions on Networking, 2022

# MEC with Network Slicing



Two different policies $P_b^*$ and $P_b^{cp}$

The proposed policy using NS achieves 2.5 times better performance.

Gain reduces when no. of WDs increases.

# Topics

- Introduction
- Architecture
- Deployments and Use Cases
- Network Slicing
- MEC in Massive IoT
- Related Research
- Conclusions

# Conclusion

- MEC is an alternative to cloud computing and offers low latency, scalability, and privacy.
- MEC enables many novel use cases due to proximity and location-awareness.
- MEC has more potential when it meets network slicing.
- The capacity of MEC can be increased by Edge-Edge and Edge-D2D collaboration.
- MEC will complement (not replace) MCC. MEC has capacity limitations.

# Conclusions

- **Deployment Obstacles**
- **How** and **When** to start deployment?
  - Which services/applications require ultra-low latency, high storage capacity, etc.?
  - How close should the edge be located to the device – at RAN, at aggregation point, inside a vehicle, etc.?
  - Single service MEC or Multiple services MEC?
- **Who** will own the edge?
  - MNOs
  - CSPs
  - Public cloud providers (sell to CSPs, sell directly to users)
  - System integrators
  - Open Ecosystem (multiple stakeholders)

# Thank you!
# Q&A

Muhammad Asif Khan, SMIEEE, PhD, CEng

**Email:** asifk@ieee.org | **Web:** https://muasifk.github.io/