Fog/Edge Computing Platform : Enabling Low-Latency Application in Next Generation Network

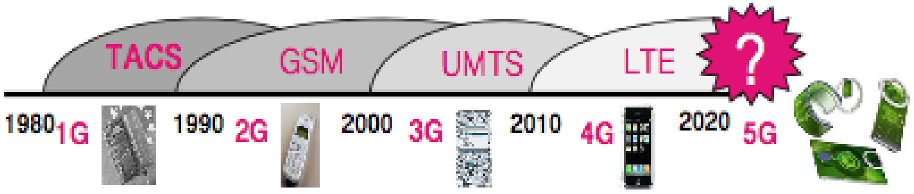
Dr. Yuan-Yao Shih

Postdoctoral Research Fellow

Research Center for Information Technology Innovation

Academia Sinica, Taiwan

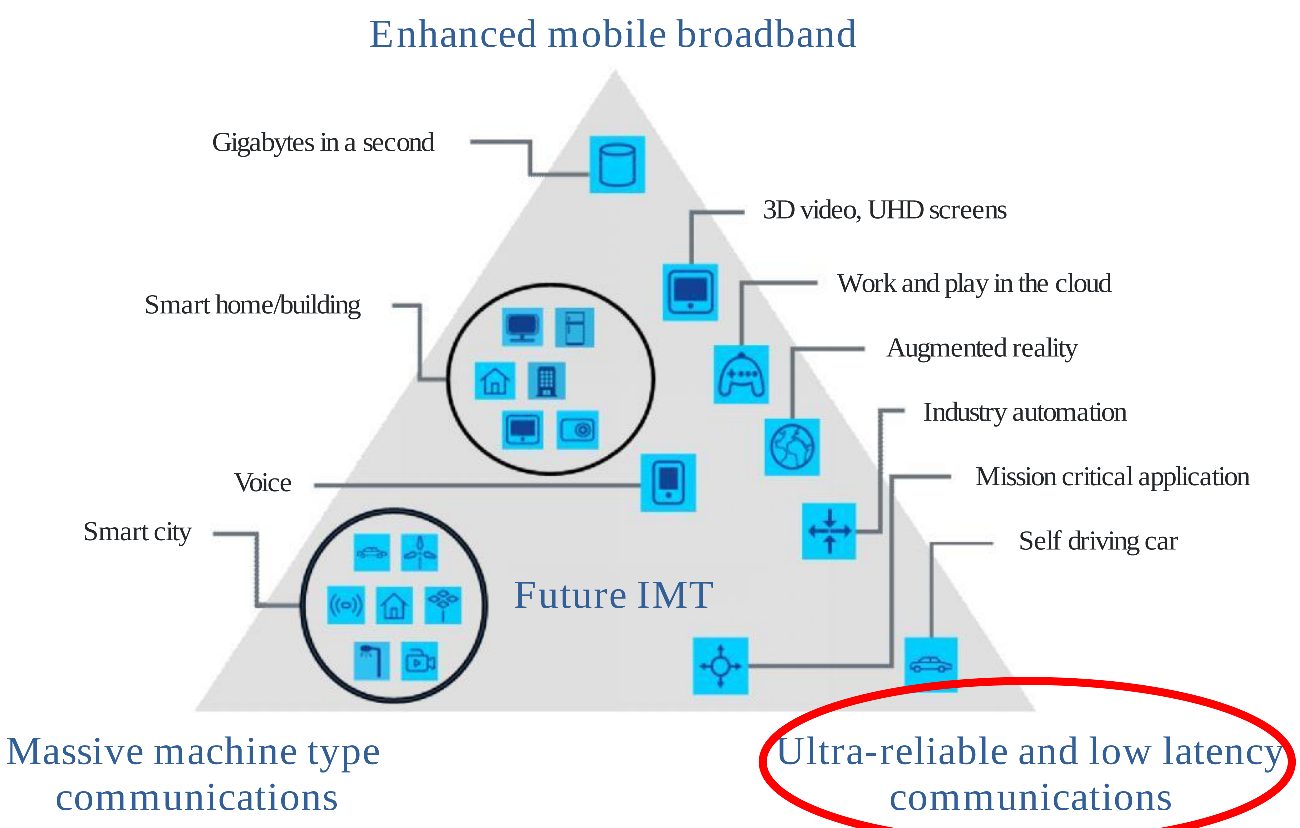
Trends for Future Wireless Comm.



- Data traffic avalanche
- Massive growth of connected devices
- Diversification of services and equipment
- Vertical markets

Vision for 5G

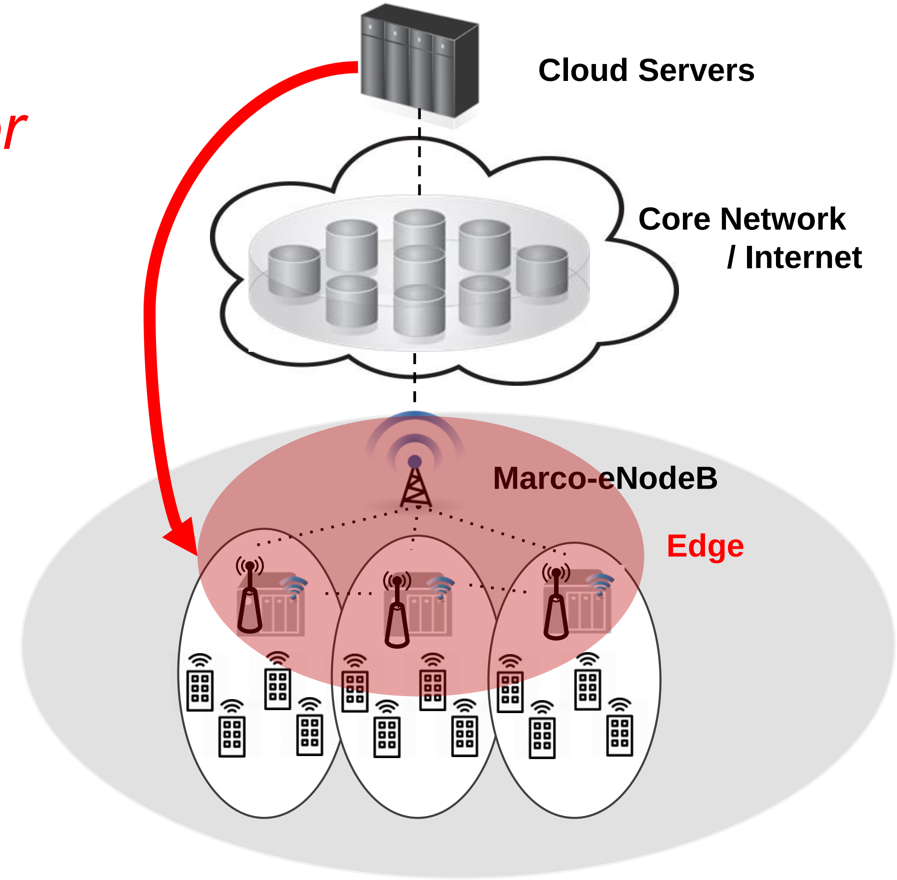
New use scenarios will emerge calling for requirement enhancement: **Mobile Broadband, Massive Connectivity, Low Latency**.



Ultra Low Latency Realization in 5G

In order to realize the latency of several ms, new technology will be required.

"Push to the edge of network for low latency"



Cloud Computing

- Centralized pooling
- Efficient resource utilization

Fog Computing

- Close to the edge
- Low latency

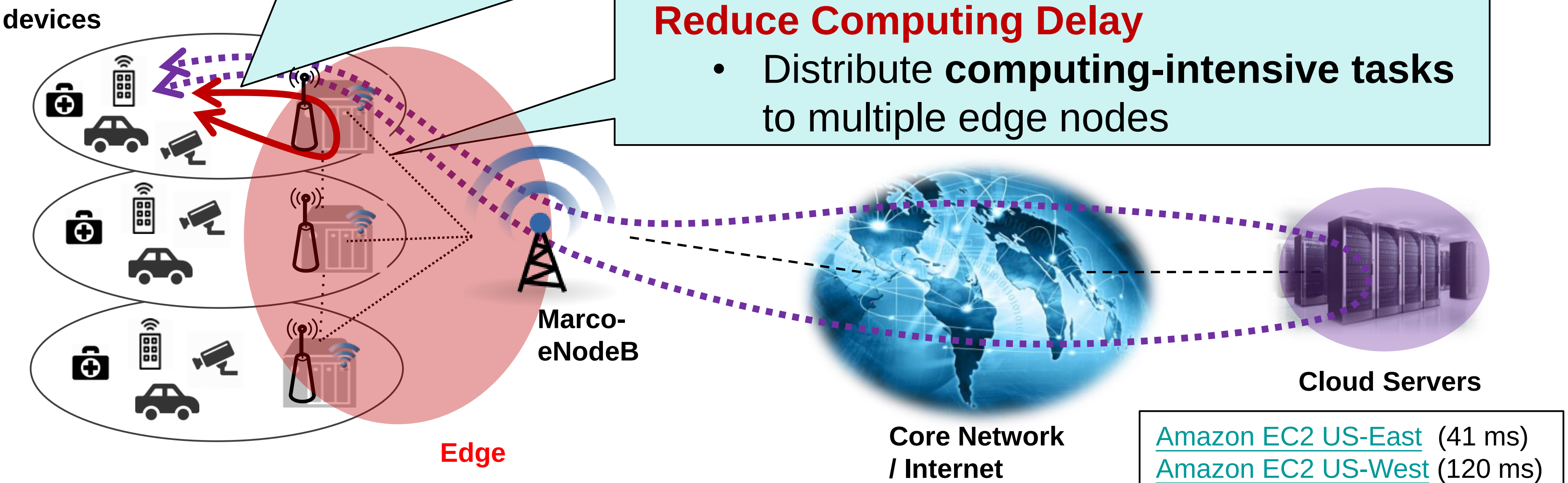
How Fog Computing Works

Reduce Communication Delay

- **WAN latency** is hard to improve
- Some applications require **bulk processing data** for computing-intensive tasks (e.g., real-time video analytics)

Reduce Computing Delay

- Distribute **computing-intensive tasks** to multiple edge nodes



Amazon EC2 US-East (41 ms)
Amazon EC2 US-West (120 ms)
Amazon EC2 EU (190ms)
Amazon EC2 Asia (320ms)

End-to-End Latency Measurement by CMU

Research on Fog/Edge

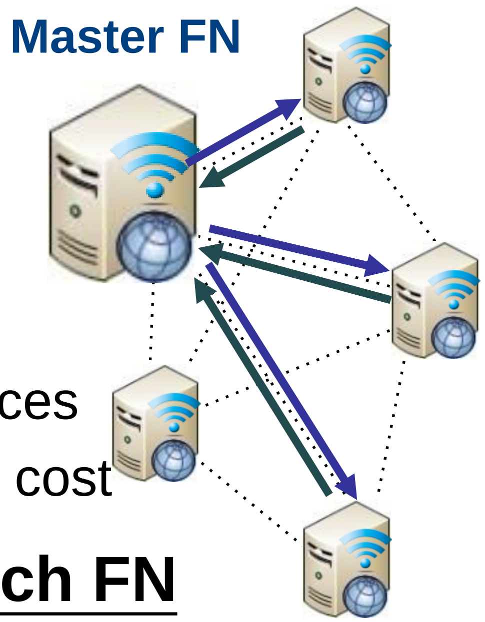
1. **Resource Management**
   - Joint design on computing and communication resource allocation
   - "Latency-Driven Cooperative Task Computing in Fog-Radio Access Networks," *IEEE ICDCS 2017*
2. **Service Provisioning**
   - Container-based virtualization for provisioning wearable applications in WiFi access points
   - "A Virtual Local-hub Solution with Function Module Sharing for Wearable Devices," *IEEE MSWiM 2016*
3. **Fog-based Platform**

R1: Challenges for Computing in Fog/Edge

- **The computing capability of an Fog node (FN) is very limited.**
  - Single FN is not capable for computing-intensive tasks.
  - Propose to do the application-layer **computing collaboratively involving multiple FNs**.
- **How to decide how many and which FNs to be involved**
  - A new type of **cost (communication/computing)-performance tradeoff** where the **temporal equivalency** of the two physically different resources needs to be built.

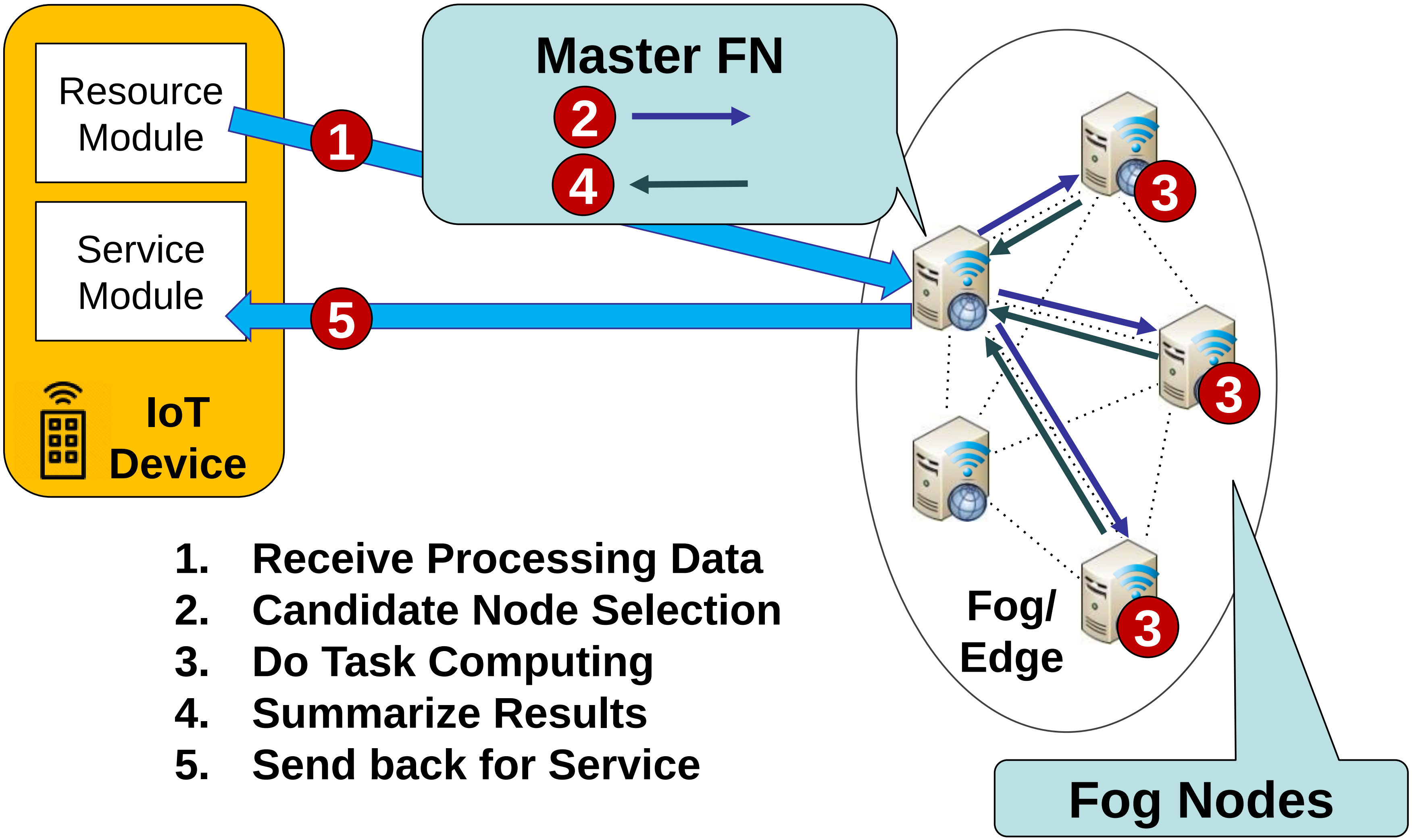
A New Type of Comm. and Comp. Tradeoff

- Need to tackle the issues considering the **tradeoff between communication and computing** in **temporal domain**
  - More FNs ► Higher computing power for all system (lower comp. delay) but lower communication resources for each FN (higher comm. delay)



- **Decide which FNs to be selected**
  - Attributes of master FN ► communication resources
  - Distances between master FN and FNs ► comm. cost
- **Decide amount of computing tasks for each FN**
  - Attributes of FNs ► computing resources
  - Loading of FNs ► computing cost

Cooperative Computing in Fog/Edge



1. Receive Processing Data
2. Candidate Node Selection
3. Do Task Computing
4. Summarize Results
5. Send back for Service

Cooperative Task Computing Operation (1/2)

- **Special case for one user**:
  - Design a *dynamic programming* approach (CTC-DP)
  - Proof of *optimal* solution for minimum service latency
  - Based on recursive formula $g(r, c, f)$ to build a DP table

$$g(r,c,f) = \begin{cases} 0, & \text{if } c = 0 \\ \infty, & \text{else if } r = 0 \text{ or } f = 0 \\ \min\limits_{\hat{r} \in [1,r], \hat{c} \in [1,c]} \Big(\max(g(r-\hat{r}, c-\hat{c}, f-1), t^{f}_{\hat{r},\hat{c}}), g(r,c,f-1) \Big), & \text{otherwise} \end{cases}$$

  - Two procedures:
    - ✓**FILL-TABLE()**: fills the DP table by $g(r, c, f)$
    - ✓**BACK-TRACE()**: selects the feasible set of FNs with cooperative tasks assignment

Cooperative Task Computing Operation (2/2)

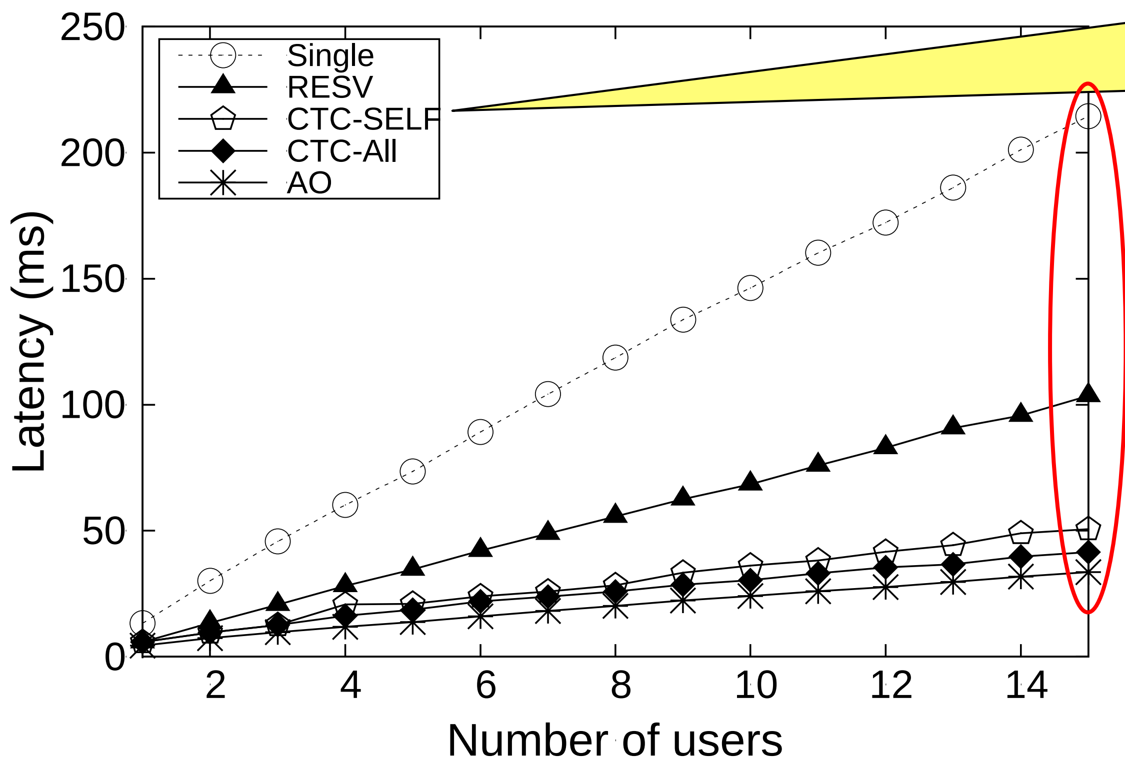
- **General case for multiple users**:
  - Design a heuristic algorithm (CTC-All)
    - ✓ Propose *one-for-all* concept to consider other's *side-effect*
  - Avoid *resource starvation* and *utilization degradation*
  - Two stages:
    - ✓ **Heterogeneous resource allocation**
      - Decide comm. resources based on processing data weight
      - **Dynamic comp. resource allocation** under distributed architecture
    - ✓ **Cooperative task computing**
      - Leverage CTC-DP with *one-for-all* concept for solving each user's cooperative task computing

Simulation Setup

- Communication considers **path loss**, **shadowing**, and **multipath fading**
- Computing ability are estimated by **ARtoolKit** [1] **Valgrind** [2]
- Frame Width: QCIF 176×144 pixels [2][3] (Encode with H.264)
- Bits/pixel: 8 bits (Gray scale)
- Max RB number: 100 (Based on **LTE specification -** *3GPP TS 36.211*)
- Data rate per RB: 9.6, 14.4, 19.2, 21.6 Kbps
- Max FN number: 20
- Platform: Intel i7 Core 2.5GHz, Dual core, 8G RAM
- Computing Power: 700 - 1700 Million Instructions/sec

[1] ARtoolKit, Available: http://artoolkit.sourceforge.net
[2] Valgrind, Available: http://valgrind.org/
[3] Video sequences, Available: http://trace.eas.asu.edu/yuv/
[4] J. Ha, K. Cho, F.A. Rojas, H.S. Yang, "Real-time scalable recognition and tracking based on the server-client model for mobile Augmented Reality", in IEEE ISVRI, Mar. 2011.

Exemplary Ultra-Low Latency Result



Single: Lower bound
RESV: Reservation idea
CTC-SELF: Selfish goal
CTC-All: Our scheme
AO: Upper bound

Fig. 1 Impacts of the number of users on total service latency.

CTC-All achieves **173ms (4.2x) less latency** than Single, **62ms (1.5x) less latency** than RESV and **9ms (24%) less latency** than CTC-SELF

Other Matrices

Single: Lower bound **RESV**: Reservation idea
CTC-SELF: Selfish goal **CTC-All**: Our scheme
AO: Upper bound

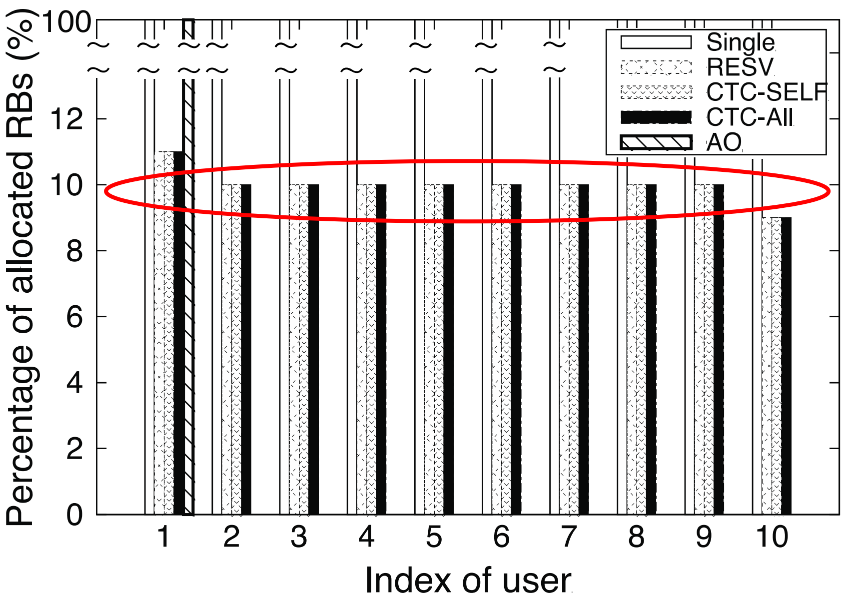


Fig. 3 Percentage of allocated RBs of each user.

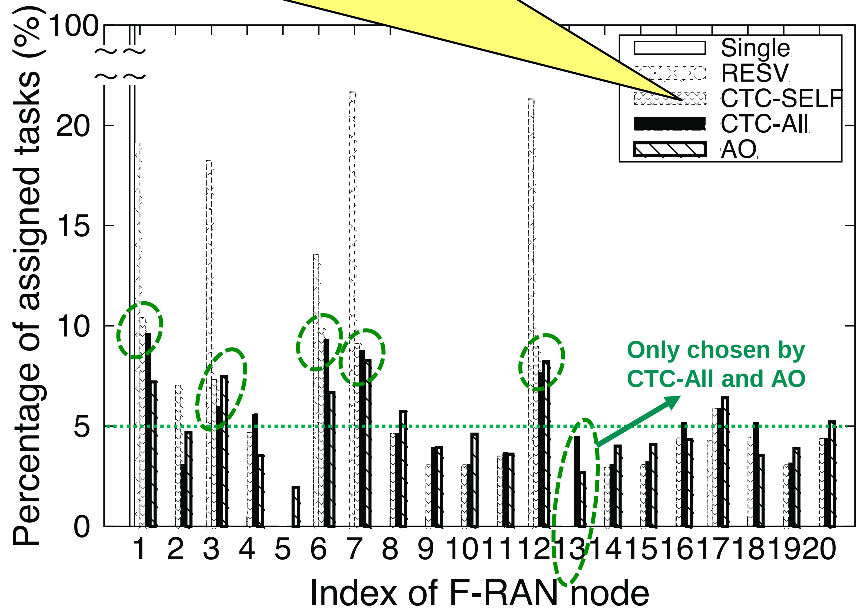


Fig. 4 Percentage of assigned tasks of each FN.

In Fig.3, ***dynamic computing resource allocation*** is the key to perform effective cooperative task computing

In Fig.4, CTC-All with *one-for-all* achieves ***load-balancing***

R2: Fog-based Wearable Applications

- Clothing or accessories worn on human body incorporating computer and advanced electronic technologies
  - Sensors
  - Processing and storage capacities
  - Wireless connectivity (BLE、Wi-Fi)
  - Display
- Characteristics
  - Light weight: easy to wear
  - Low power consumption

Local-hub

- Usually a smart-phone or tablet, installed with applications related to wearable devices
- Wearable devices are connected with a local-hub via low power wireless technologies, e.g., BLE

Inconvenience of Physical Local-hub

- Wearable devices are useless if local-hub is not nearby, for example,
  - Working out in a gym
  - Swimming in a pool
- Local-hub functionalities drawdown the battery of smart phone
- Current solutions
  - Google: Android Wear **Cloudsync**
  - Apple: **Compatible Wi-Fi** for Apple Watch

Limitation of Current Wi-Fi Solution

- Long response time
  - Raw data traveling time
    - Among wearable device, cloud, and local-hub over the Internet
    - Pre-processing of the raw data should be done on local-hub
  - Indirect data exchange
    - Cloudsync server intermediates data exchanged between wearable device and local-hub
- Shortcoming
  - Poor user experience (waiting time)
  - More power consumption (screen-on time)

Concept of Virtual Local-hub (VLH)

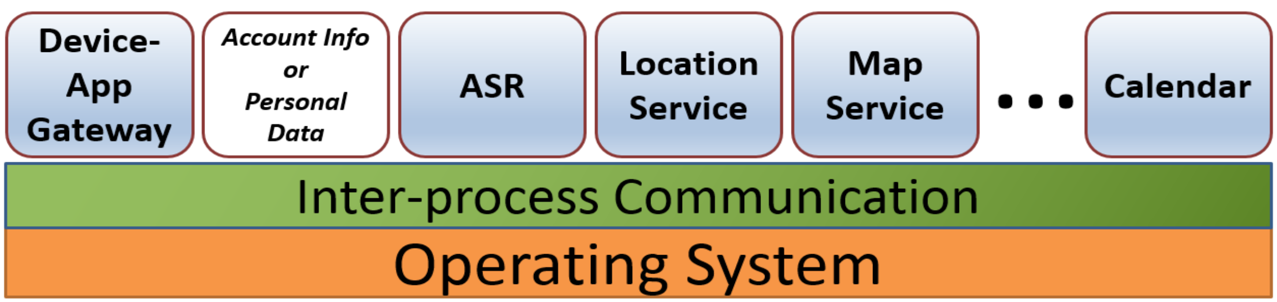
- Virtual Local-hub (VLH)
  - Wearable devices can utilize network edge nodes nearby to serve as their local-hub instead of smartphones
- Basic ideas make VLH to be practicable
  - Fog computing
  - Virtualization technology
- Intuitive idea of VLH
  - Virtualize all applications of local-hub in a smartphone as a virtual machine (VM)
  - Migrate the whole VM to edge nodes (e.g., Wi-Fi AP) nearby the user

Issues of VM Migration

- Long migration time of whole VM
  - Size of a VM is quite large (about hundreds of MBs)
- Capacity limitation of a Wi-Fi AP
  - Processing/storage resources are restricted on an AP
  - A Wi-Fi AP may only accommodate few VMs
- Not a cost-effective solution

VLH System Design

- Idea 1: Fog Computing realized by a group of Wi-Fi APs
  - Wi-Fi APs can connected with each other on a LAN
- Idea 2: Container-based Virtualization
  - Modular programming environment for mobile APP
    - Developers can adopt existing function modules to build the applications for wearable devices
  - To virtualize function modules as containers



Objective

- To mitigate the side-effect of function module sharing
  - To Minimize the total bandwidth consumption of edge network
- Challenges
  - How many FM instances should be executed on VLH network?
    - **Resources usage** decision
  - How to allocate these FM instances?
    - **Migration** decisions
    - **Allocation** decisions
  - How to share these FM instances?
    - **Call graph mapping** decisions

Proposed Algorithm

- **Nearest Serving Node (NSN) Algorithm** (Greedy-based)
  - Key Idea: A FM instance should serve those requests as near as possible

Resources Usage Decision → **FM Instances Allocating Procedure**

Choose **the least FM instances** for allocation based on sharing limit

For each FM instance

- Try **every node** on edge network
  - Migration bandwidth consumption
  - Bandwidth consumption of serving these SRs
- Choose the **least bandwidth consumption** one

Performance Evaluation

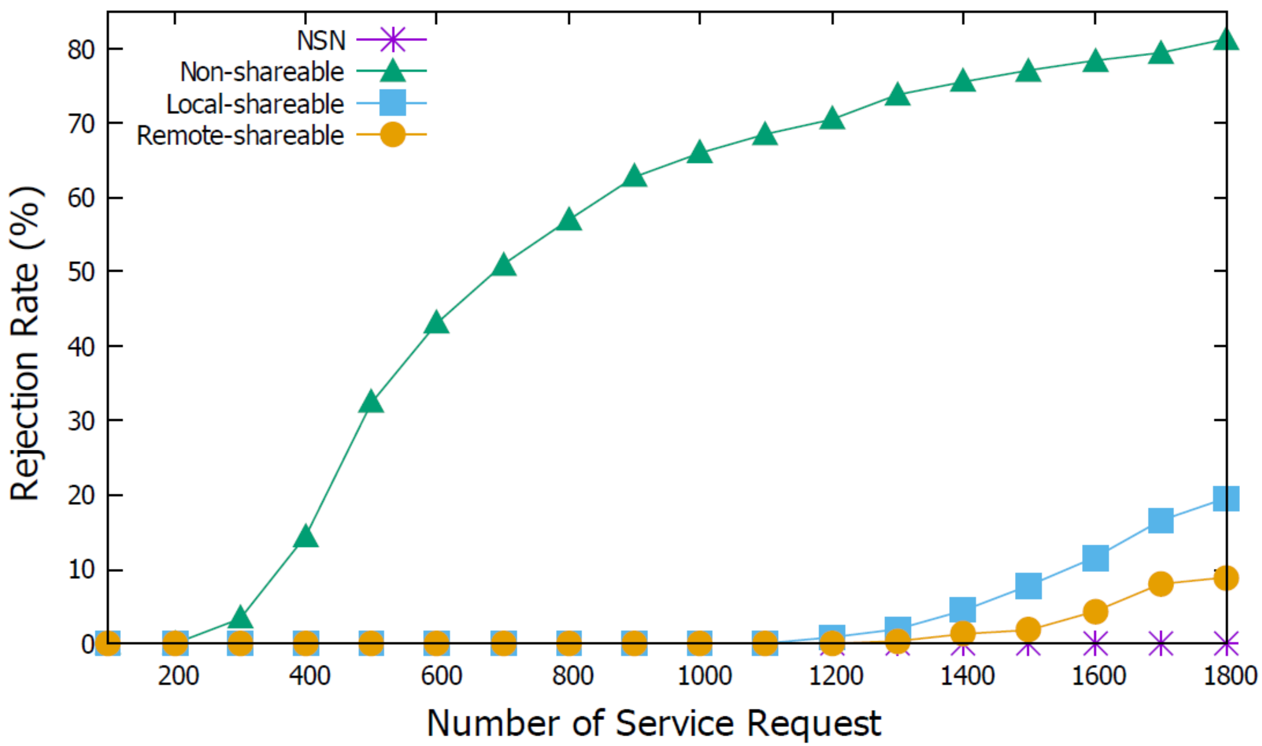
- Simulation Setup
  - Number of Wi-Fi APs: 100
  - Available bandwidth capacity: 1 Gbps
  - Available computing capacity: 1000
  - Number of function module (FM) types: 20
  - Bandwidth requirement of FM types: 1-150 Kbps
  - Computing requirement of FM types: 5-100
  - Package size of FM types: 1-15 MB
  - Number of call graph types: 20
  - Number of service requests: 500

Performance Evaluation

- We conduct two kinds of comparison
  - Comparison of Different Sharing Strategies
    - To assess the impact of different function module sharing strategies on the rejection rate
      - Non-shareable
      - Local-shareable
      - Remote-shareable
  - Comparison of Different Allocation Strategies
    - To investigate the performance of total bandwidth consumption
      - First In First Out (FIFO)
      - Random

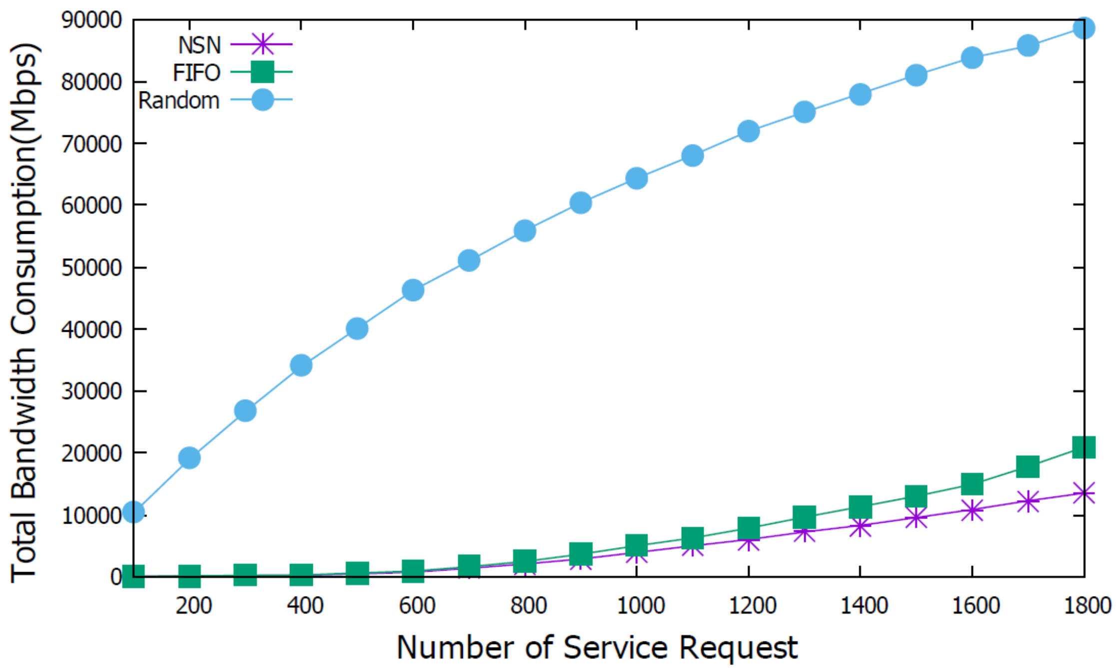
Comparison of Sharing Strategies

- Non-shareable suffers from high rejection rate
  - Up to 80% service requests cannot be accommodated
- Remote FM sharing can reduce rejection rate significantly



Comparison of Allocation Strategies

- Impact of the number of service requests (migration occurs due to limited storage size)



(a) Total bandwidth consumption

R3: OmniEyes: Fog-based Video Management Platform

- The generation of video data has started a paradigm shift from the **content provider** to **individuals** and now the **“things”**

12B hours of Netflix in Q4 2015 and 2015 revenue sales: 6.78B USD

Snapchat: 7B video views per day
Periscope: 100m live broadcasts in 10 months

Video → **Video & Sharing** → **Video & Sharing + Mobility & Analytics**

We want to become the "Mobile Video" King of the physical world

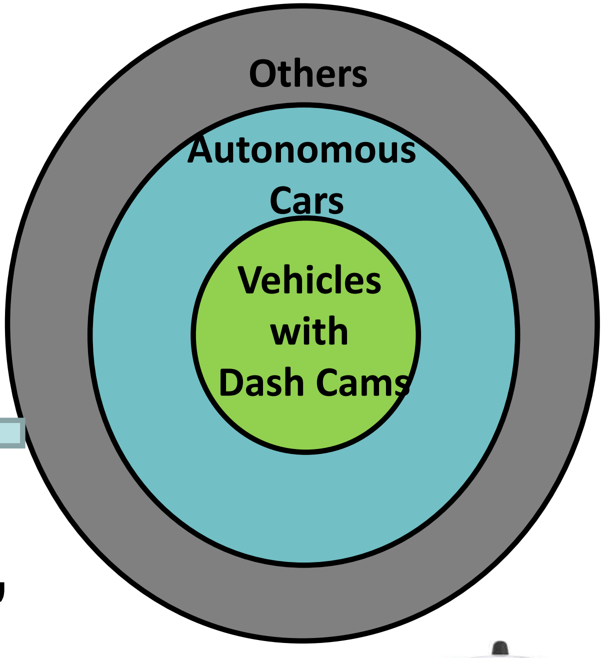
To Change the way people explore the physical world
with our **omnipresent videos**

New ways of Searching, Driving, and Tracking
New ways of Mobile Advertisement and Auto Insurance

Our OmniEyes Platform

1. Street View Live
2. Harperscope
3. Time Machine

Fog Based Video Platform



A platform to share, to fuse, to analyze, and to render location-dependent, video data + information

Conclusion

- **Low latency** is required by many existing and new usage scenarios for future communications.
- Fog computing is the key to realize low-latency communications.
  - It also makes ISP/carrier turn from dump-pipe into smart-pipe.
  - Orchestration of fog and cloud
- There will be huge research and business opportunities following this direction.


thank you
danke
рахмат
謝謝
ngiyabonga
teşekkür ederim
Баярлалаа
спасибо
mersi
kia ora
barka
welalin
tack
spas
dank je
misaotra
matondo
paldies
grazzi
mahalo
gracias
tapadh leat
faafetai lava
vinaka
спасибі
blagodaram
kiitos
dankie
dhanyavad
nanni
nandri
mauruuru
hvala
bedankt
köszönöm
gràcie
bayarlalaa
enkosi
dziękuję
akun
dankon
ačiū
хвала
asante
manana
obrigada
murakoze
tenki
djiere dieuf
tau
mochchakkeram
mamnun
дякую
chokrane
go raibh maith agat
chnorakaloutioun
gratias ago
gràcies
sulpáy
taiku
sobodi
děkuji
obrigado
sagolun
sukriya
kop khun krap
arigatô
takk
dakujem
trugarez
mèsi
didi madloba
najis tuke
ありがとう
tanemirt
grazie
rahmet
kam sah hamnida
rahmat
terima kasih
diolch
dhanyavadagalu
shukriya
mercé
мерси
তোমাকে ধন্যবাদ
감사합니다
xièxie
ευχαριστώ
merci