Balancing Computing and Networking in Autonomous Edge Clouds

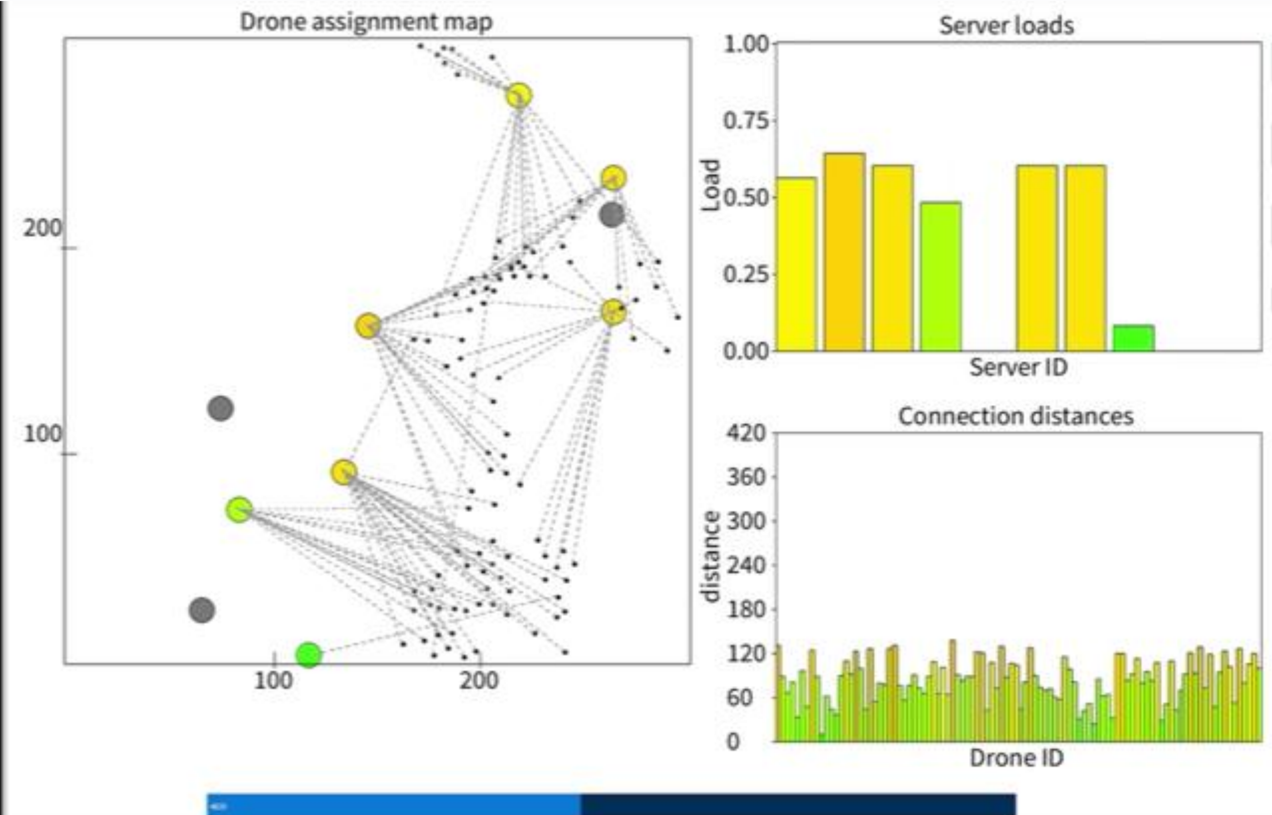
Shinnosuke Masuda[1], Tsuyoshi Hasegawa[1], Kazuyuki Shudo[1], Kenjiro Cho[2]

[1]Kyoto University, [2]IIJ Research Laboratory

増田 真之介[1], 長谷川 毅[1], 首藤 一幸[1], 長 健二朗[2]

[1]京都大学, [2]IIJ 技術研究所

Balancing Computing and Networking in Autonomous Edge Clouds



Shinnosuke Masuda[1] 増田 真之介[1]

Tsuyoshi Hasegawa[1] 長谷川 毅[1]

Kazuyuki Shudo[1] 首藤 一幸[1]

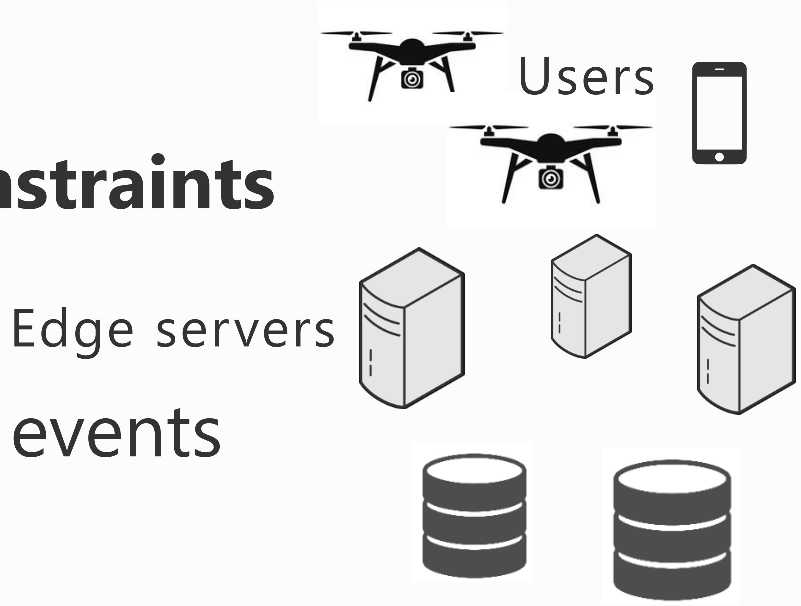
Kenjiro Cho[2] 長 健二朗[2]

[1]Kyoto University,
[2]IIJ Research Laboratory

[1]京都大学,
[2]IIJ 技術研究所

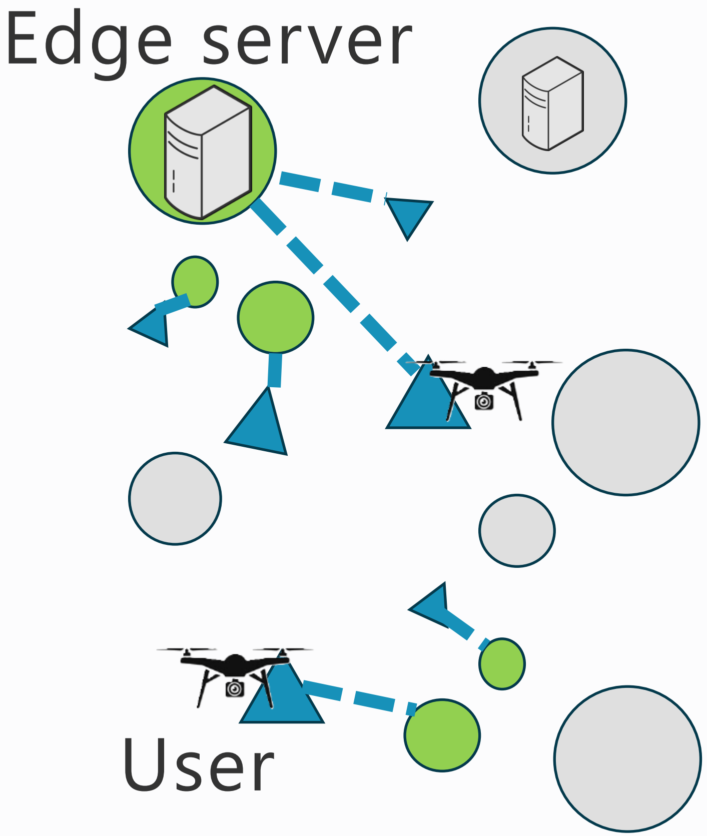
Background : Emerging edge computing

- **Diverse** and **geographically scattered** **edge** computing resource can be utilized as part of larger cloud services
  - Pc servers and micro-datacenters
  - Constructed by micro services
- Dynamic resource allocation **while multiple constraints**
  - Computing load, network latency, …
- Needs to be resilient in coping with unexpected events
  - **Decentralized** and **autonomous**



Background : Cloud Morphing [Cho2023]

- Autonomous resources allocation **using pseudo cost functions**
  - Convex function to achieve moderate loads
- Find the best edge node for a given job
  - By a simple cost-minimizing job allocation method
- The cost of a resource and allocation **dynamically changes** along computing load
  - To avoid over-concentration



[Cho2023] K. Cho and J. Baffier, “An Autonomous Resource Management Model towards **Cloud Morphing**,” in ACM **EdgeSys’23**, 2023, p. 7–12. [Online]. Available: https://doi.org/10.1145/3578354.3592864

Background : Hard and soft Constraints

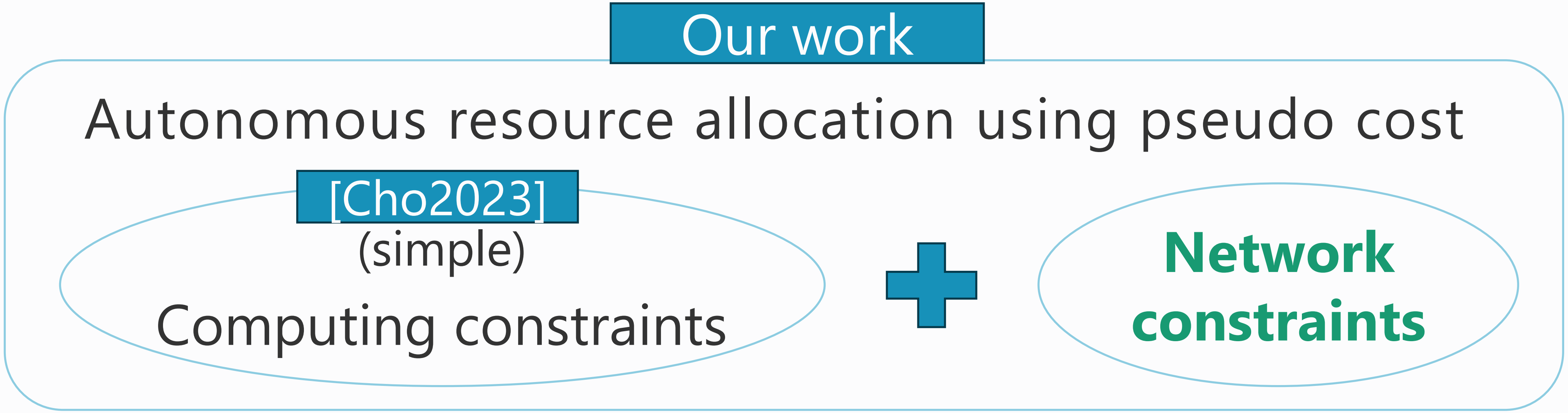
Constraints in the resource allocation

- **Hard constraints** : the system has to always adhere
  - Capacity limit
- **Soft constraints** : the system should maintain as much as possible
  - Computing load
  - Network distance
  - Fair Network bandwidth allocation
  - …

System designers **determine the priorities of these constraints**
based on the objectives they wish the system to achieve

Goals

- Autonomous resource allocation
  - Using **pseudo cost function** in Cloud Morphing [Cho2023]
  - For future edge computing
- Consider **multiple constraints** on **computing** and **networking**
- Evaluate how computation and communication are balanced with multiple constraints **through simulations**



Our work

Autonomous resource allocation using pseudo cost

[Cho2023]

(simple)

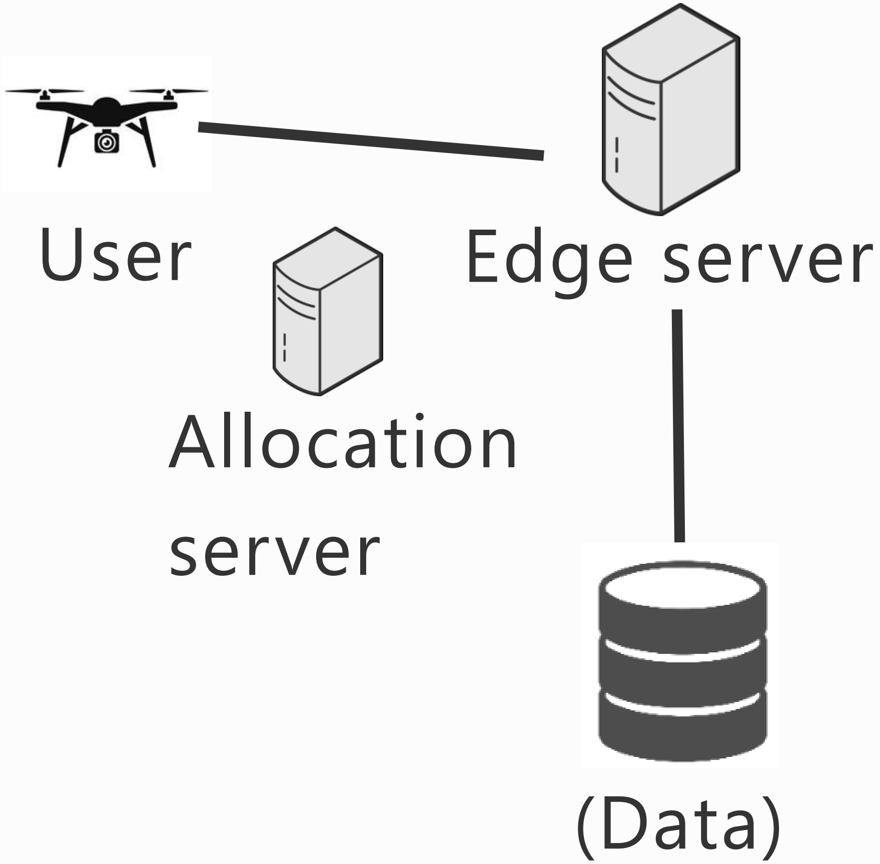
Computing constraints

+

Network constraints

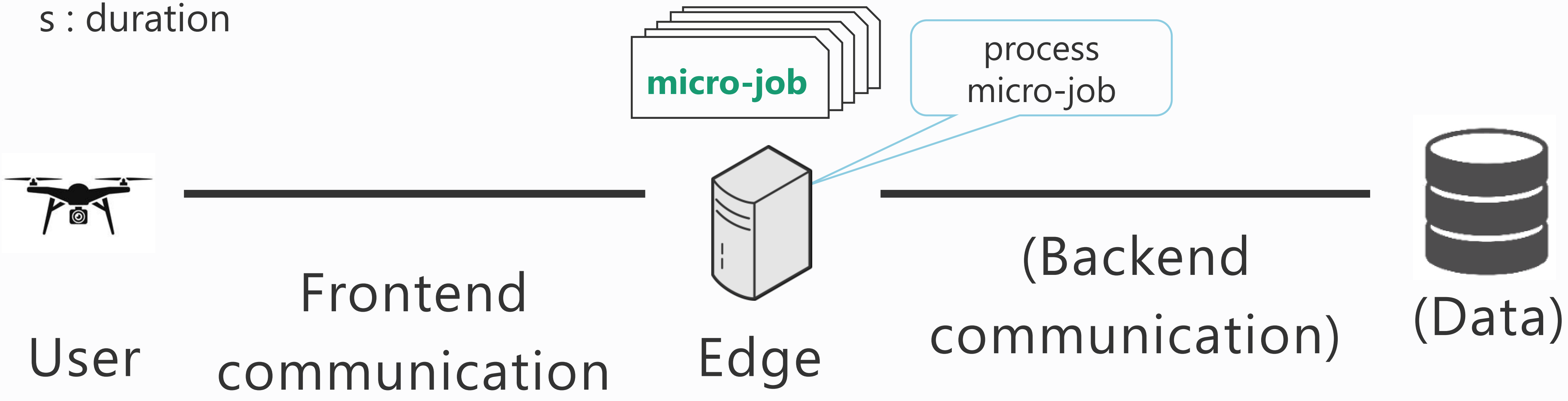
Model : System

- **User**
  - **Requests a service** using a remote data **to a nearby allocation server**
- **Edge server**
- **Allocation server**
  - Creates a series of **micro-jobs**
  - Calculate **pseudo-cost** to host jobs
  - Assigns job to **the cost-minimizing node**
- **(Data)**



Model : Micro-job

- **Requested by user** and **processed by edge server**, using data object in data center
- Short-lived and **independent each other**
- **Micro-job** definition : J(p,q,r,s)
  - p : required computational units
  - q : frontend communication amount
  - r : backend communication amount
  - s : duration



micro-job

process micro-job

User

Frontend communication

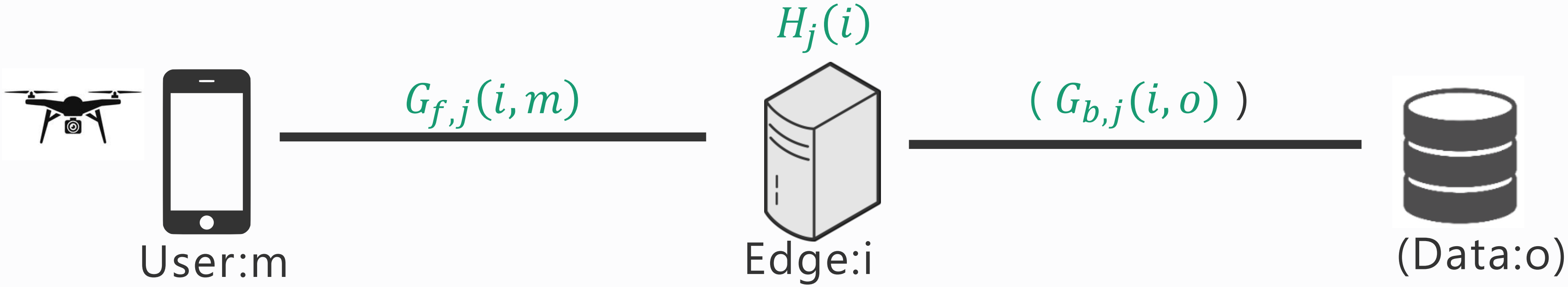
Edge

(Backend communication)

(Data)

Model : Pseudo cost

- **Pseudo Cost E** to host micro-job j at edge server i for user m and data o
  - sum of **computing cost $\boldsymbol{H}$** and **communication cost $\boldsymbol{G}$**
  - $\boldsymbol{E_j(i)} = H_j(i) + G_j(i,m,o)$
    - $G_j(i,m,o) = G_{f,j}(i,m) + G_{b,j}(i,o)$
- allocation server selects edge that minimizes the pseudo cost E
  - $argmin_i\ E_j(i)$



Constraints for pseudo cost function

- A pseudo cost function is to be defined according to each specific system.
    - Important constraints depend.

- Constraints examples :

computing

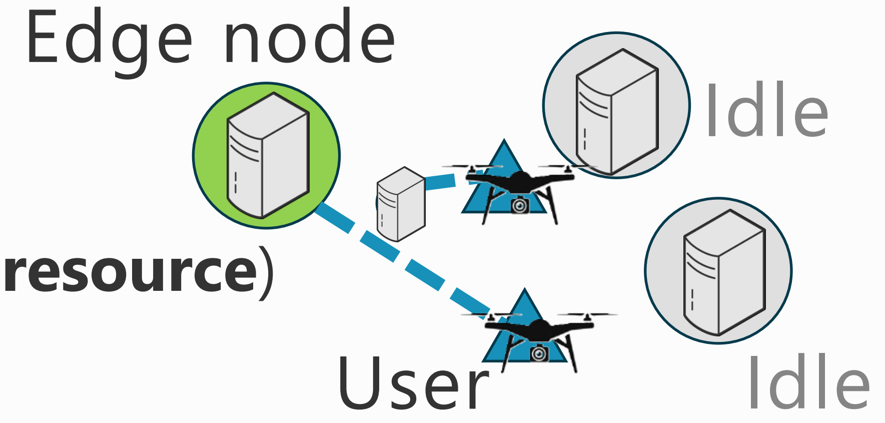
- Capacity limit
- Load balancing
- Idle-resource pooling
- CO2 emission
- Monetary cost
- …

networking

- Communication distance
- Latency
- Bandwidth
- …

Example : Convex func for idle-resource pooling [Cho2023]

- **Capacity limit** (**hard constraints**)
- **Idle-resource pooling**(**soft constraints**)
  - Keep unnecessary servers in a standby mode(**idle resource**)
  - For energy saving

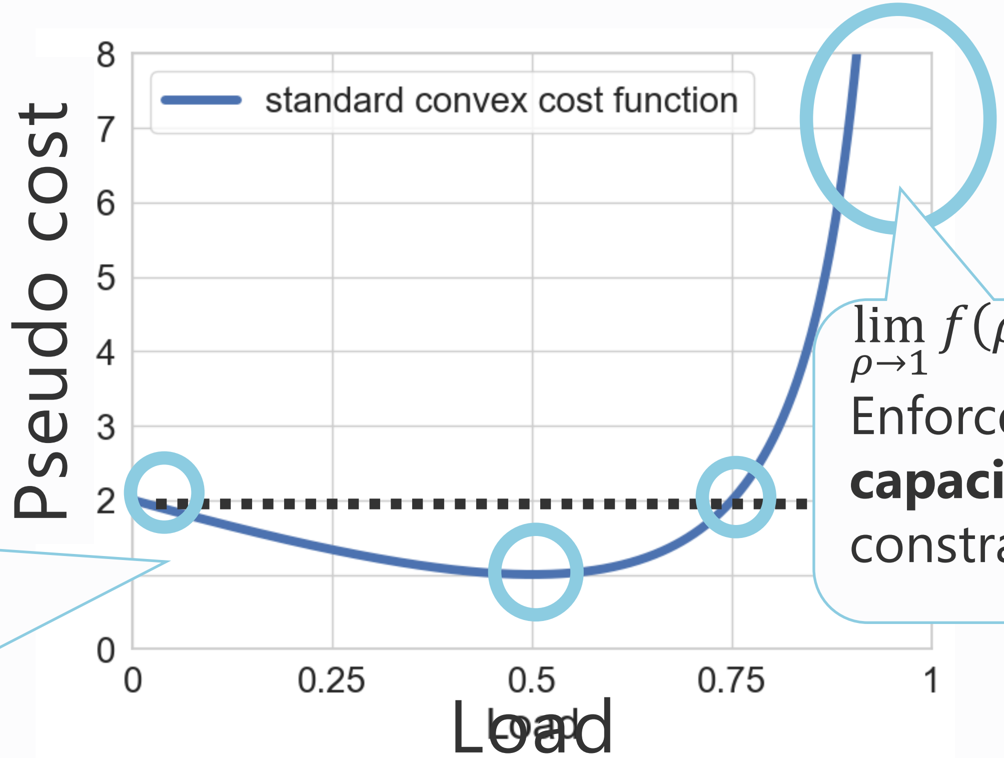


- **Computational cost func**

H = $f(\rho) = \frac{(2\rho-1)^2}{1-\rho} + 1$ ρ: edge load

$$\min f(\rho) = f(0.5)\,,\ f(0) = f(0.75)$$

- Server with zero load (Idle servers) are pooled **until active servers reach a load of 0.75**.
- The load on active servers is maintained between **0.5 and 0.75.**



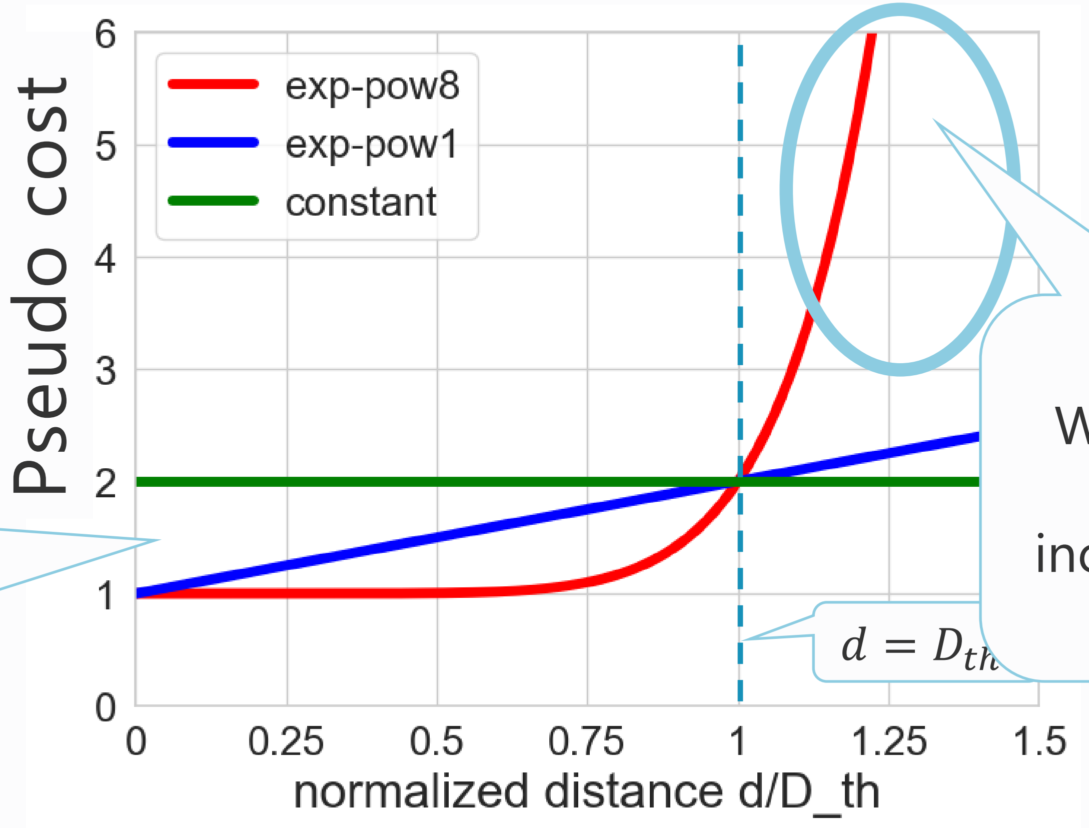
$$\lim_{\rho \to 1} f(\rho) = \infty$$

Enforce the **capacity limit** constraints

Example : Pseudo cost functions for communication

- Constraints : Keep frontend communication distance below pre-defined threshold **(soft constraints)**

- **Communication cost func**: $G(H) = g(d_{i,m}) = \left(\frac{d_{i,m}}{D_{th}}\right)^{n} + 1$
  - $d_{i,m}$: frontend distance
  - Ignore backend communication
  - D_{th} : soft distance threshold
  - n : weight



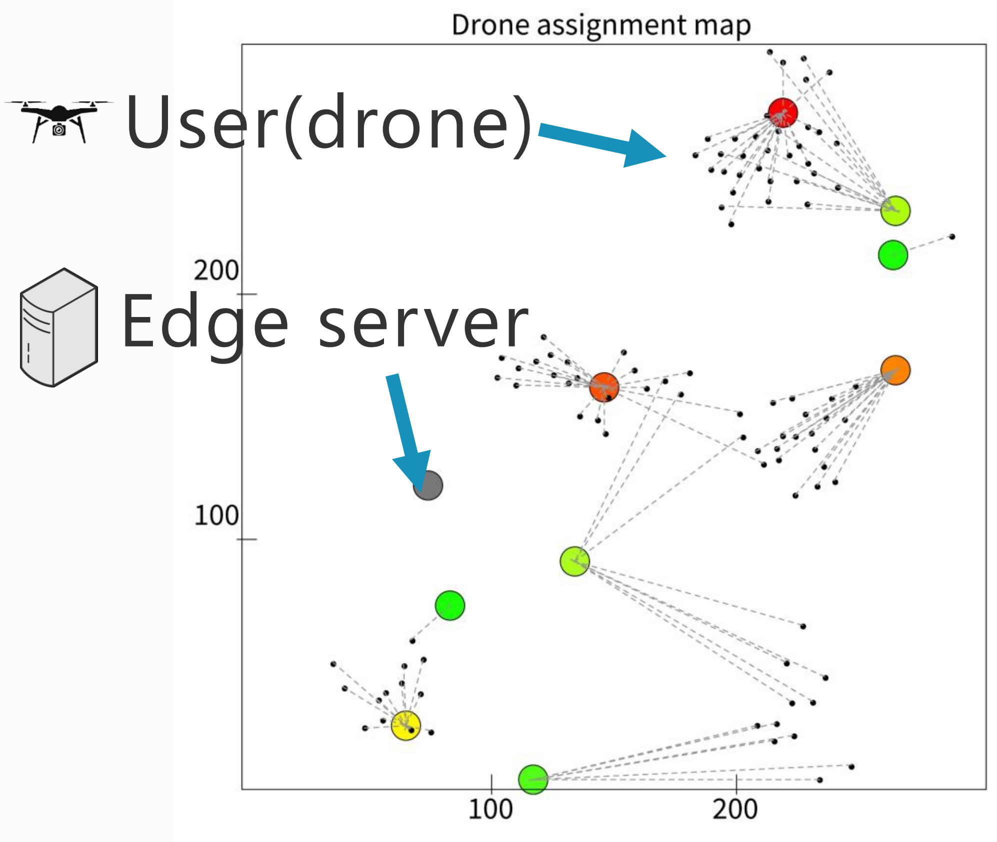
When large n, $d \geq D_{th}$ increase rapidly

Increases as the distance d grows

Micro-jobs are assigned to the nearest possible edge

Experiments of resource allocation

- **Objective**
  - Confirm that **hard constraints** are always satisfied
  - Observe that how **soft constraints** work in incorporating with other constraints
    - Computational
    - Communication
- **Method : conduct simulations**
  - Flock of **drones** move around on a square area
  - Drones are dynamically assigned to **edge servers**



Visualization of Simulation

Experiment : Constraints

Assign drones to edge servers to satisfy following constraints

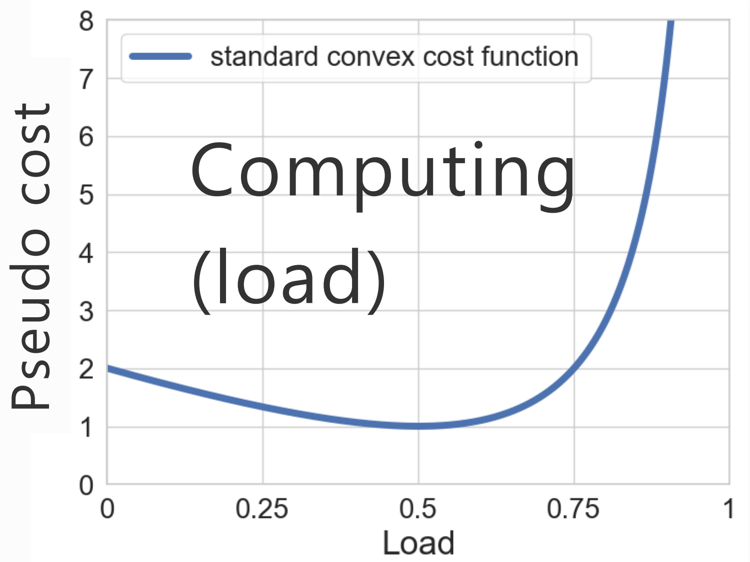
- **Computational constraints**
  - **Capacity limit**
    - Each edge server can handle up to a limited number of drones at a time **(hard constraints)**
  - **Idle-resource pooling**
    - Pool idle servers as many as possible **(soft constraints)**

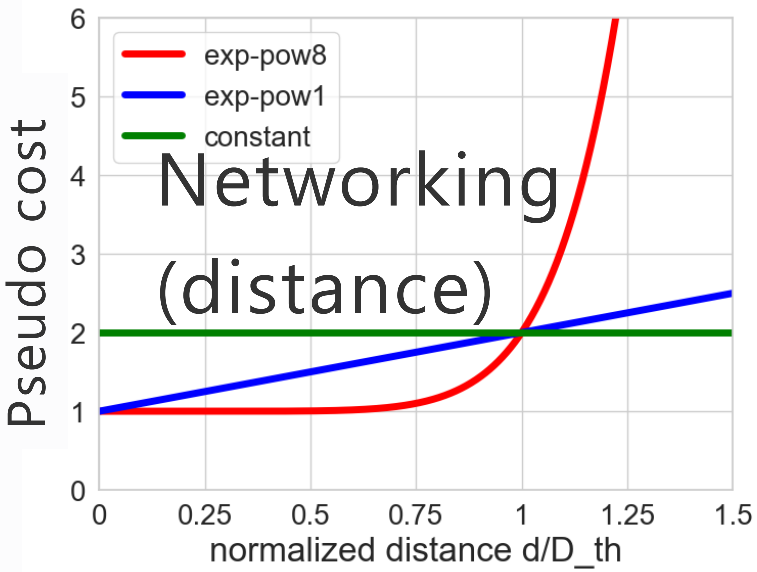
- **Communication constraints**
  - **Frontend distance**
    - Keep frontend communication distance below pre-defined threshold **(soft constraints)**

Experiment : Pseudo cost functions

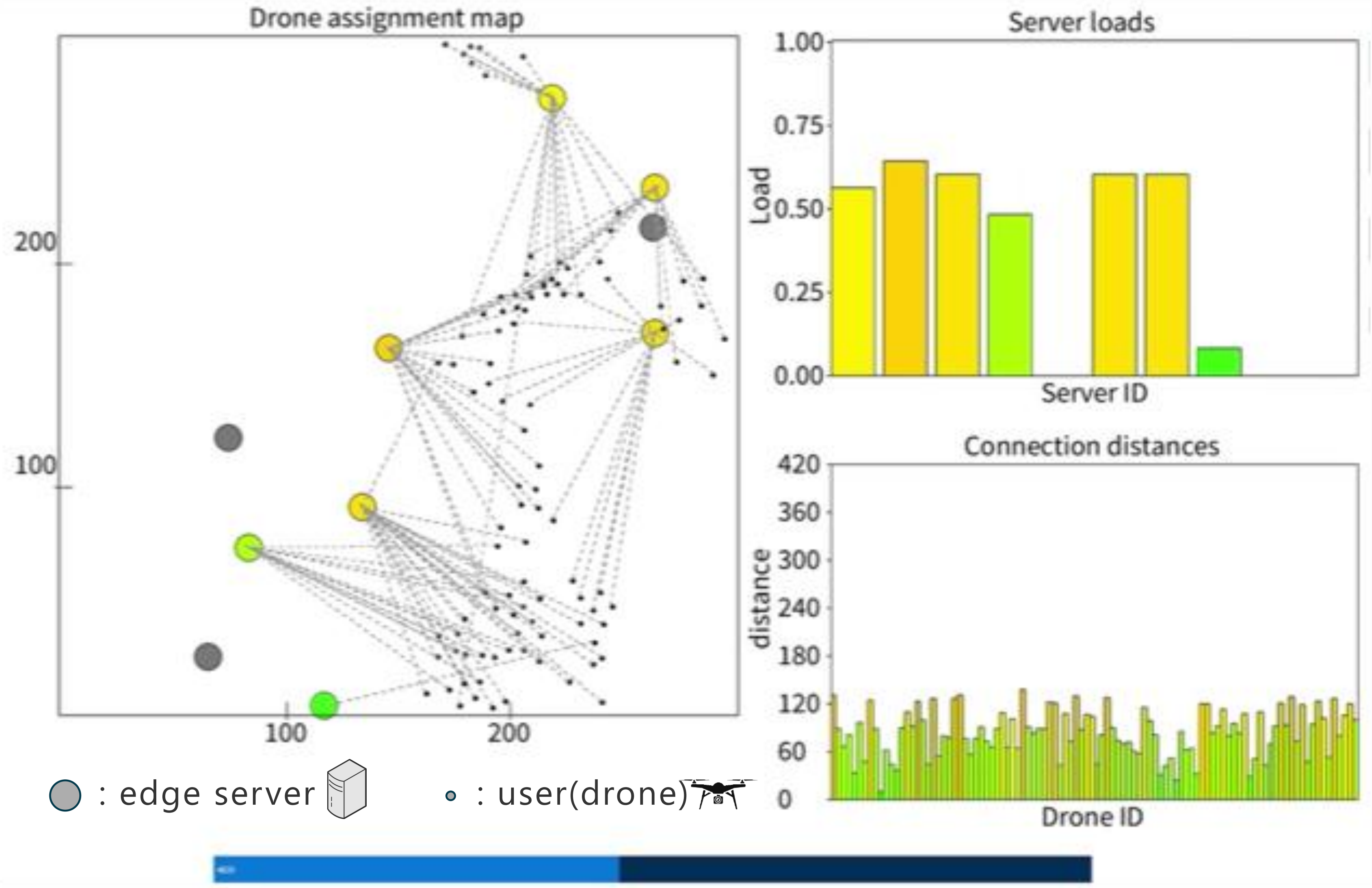
Pseudo Cost $E_j(i) = H_j(i) + G_j(i, m, o)$

- H : **Computational Cost for capacity limit and idle-resource pooling**
  - Standard convex function
    - $\mathrm{H} = \mathrm{f}(\rho) = \frac{(2\rho-1)^2}{1-\rho} + 1$
    - ρ : edge server load
    - referred to as **"convex"**
- G : **Communication Cost for frontend distance**
  - Standard exponential function
    - $G = g(d) = \left(\frac{d}{D_{th}}\right)^n + 1$
    - d : distance between drone and edge server
    - Dth : soft distance threshold
    - n : weight, $n = 1, 8$
    - referred to as **"exp-pow 1", "exp-pow 8"**





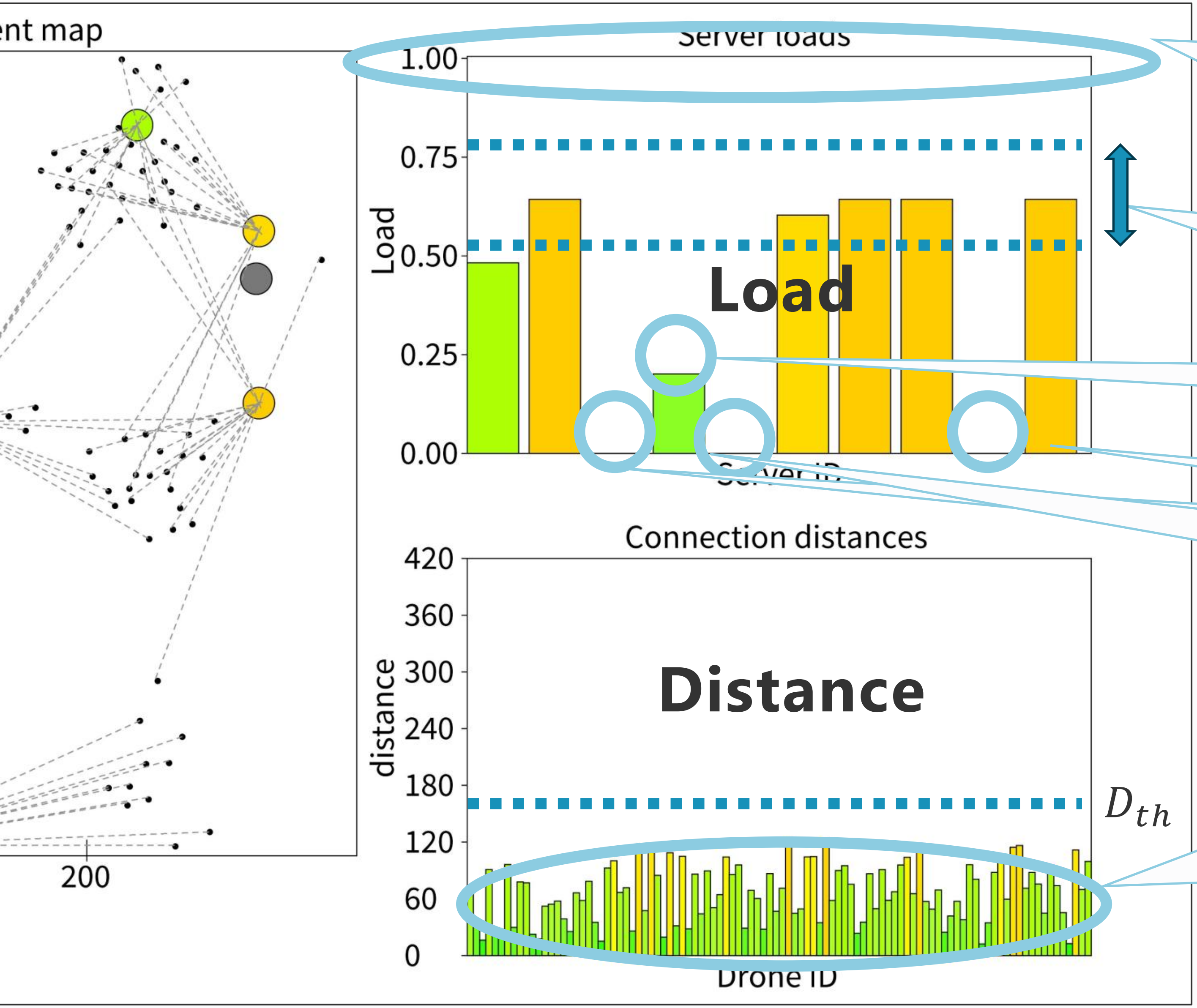
Experiment : Simulation visualization



Settings

number of drones	100
number of edge servers	10
max number of allocatable drones per server	25
map size	300 * 300
soft distance threshold (D_{th})	150

Experiment : Observations from the visualization



No server reaches a load of 1

- Capacity limit (hard constraint)

Within the target load range [0.5,0.75]

Soft constraints are not always satisfied

Servers with zero load

- Idle-resource pooling

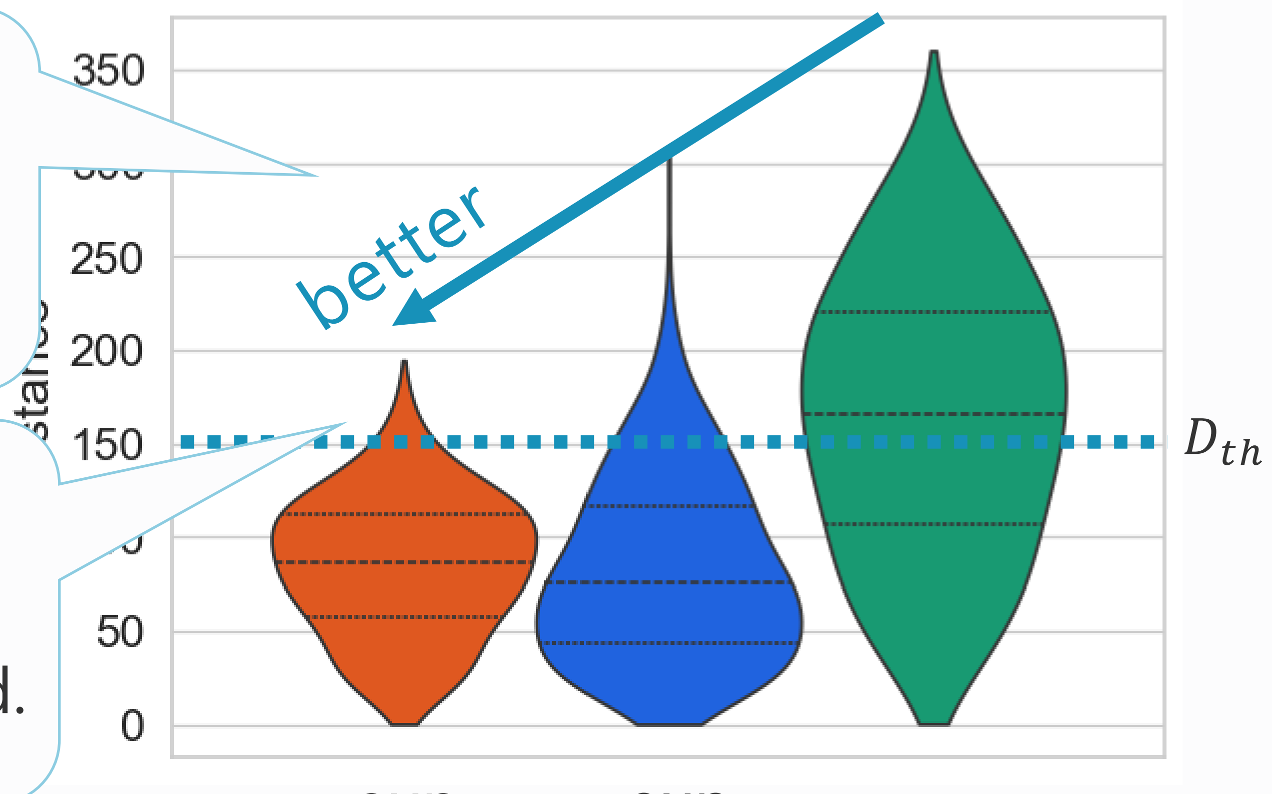
Shorter than the soft distance threshold D_{th}

Experiment : analysis for communication constraint

Plot of the **assignment distance** over the entire simulation period

Using a **strict** communication cost function results in **shorter** assignment distances

A small proportion of assignments exceeding the soft distance threshold.



communication cost functions : exp pow8 | exp pow1 | constant

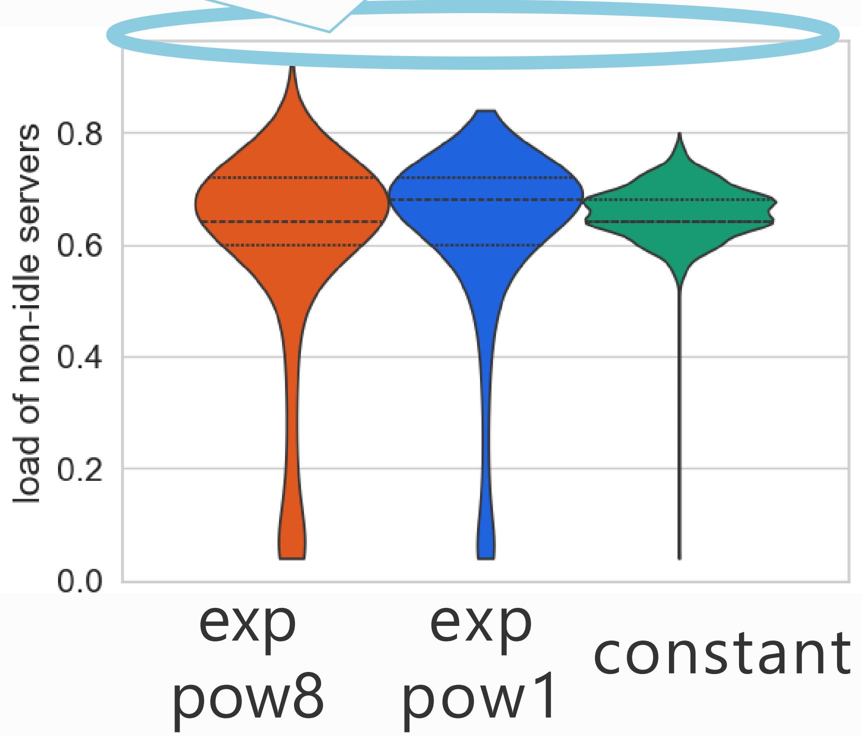
strict ⟺ weak

Experiment : analysis for computational constraint

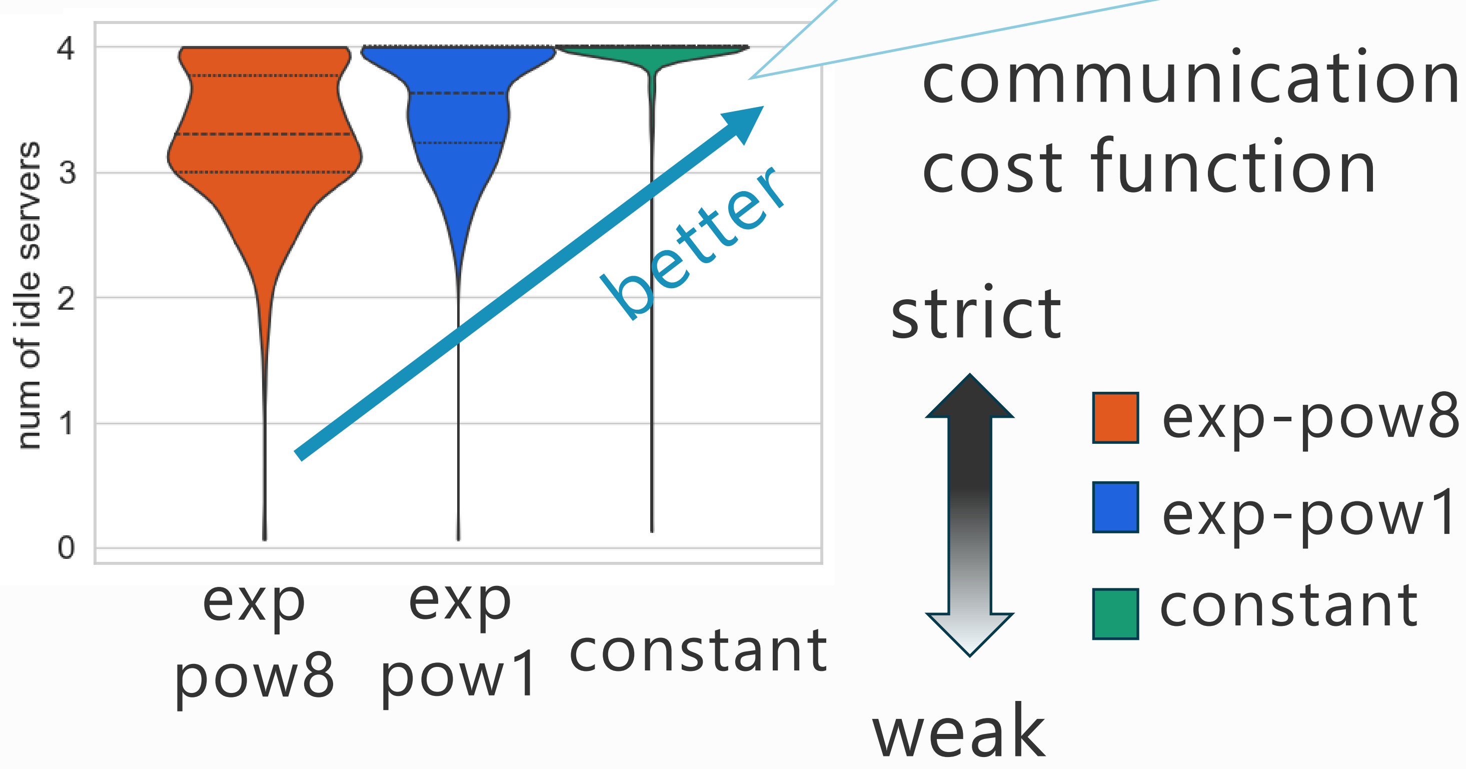
Load of active servers and **num of idle-servers** over the entire simulation period

The capacity limit constraint **(hard constraint)** is **always satisfied**

trade-off between idle-server num and communication cost function strictness



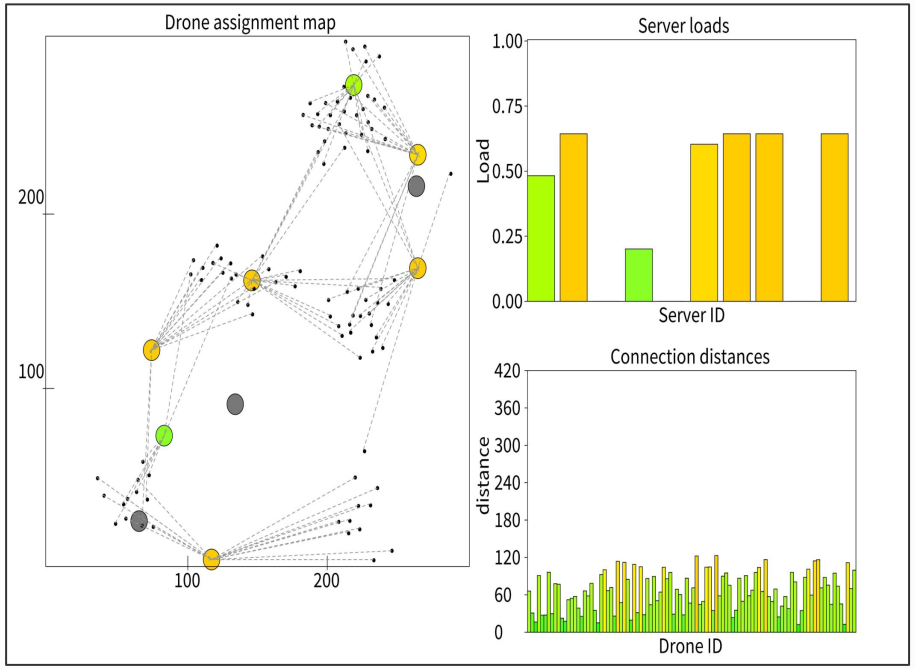
load of active servers



num of idle-servers

Experiment : Summary

- Confirmed that both computational and communication constraints can be maintained
  - The chosen cost functions **automatically balance** computation and networking
  - **Hard constraints** are **always satisfied**
  - **Soft constraints** involve **trade-offs** depending on which constraint is prioritized



Conclusion

- Edge computing in the near future would utilize flexible micro-services, leveraging **diverse and geographically scattered edge** computing resources.
- We investigate autonomous resource allocation using **pseudo cost functions** considering both **computing** and **networking** constraints.
- Through simulation
  - The chosen cost functions **automatically balance** computation and networking
  - **Trade-offs** in balancing computational and communication constraints