Towards Elasticity in Heterogeneous Edge-dense Environments

Lei Huang, Zhiying Liang, Nikhil Sreekumar, Sumanth Kaushik, Abhishek Chandra, Jon Weissman

huan1397@umn.edu

University of Minnesota
Driven to Discover®

Emerging applications enabled by edge

AR/VR
Wearable cognitive assistance
Autonomous vehicle/drone
Interactive gaming … …




Resource-constrained user-end devices

LAN → Geo-proximate locations → Co-located with 5G base station → RTT propagation delay: 1-25ms

offload computation →

← low-latency response

Edge server
(micro data center)

Existing edge infrastructures

Are existing edge resources **sufficient** to support scalable latency-sensitive computation offloading?

AWS Local Zones

Azure Stack

Answer: **No**. They are limited by geo-distribution, sub-optimal performance, high expanses, and scaling capacity.

Wavelength | 5G

Azure Edge Zones

Google Cloud Anthos

Outposts

Public edge cloud	Only available at major metropolitan areas (limited point-of-presence)
	Delivered latency performance is less satisfactory (shown later)

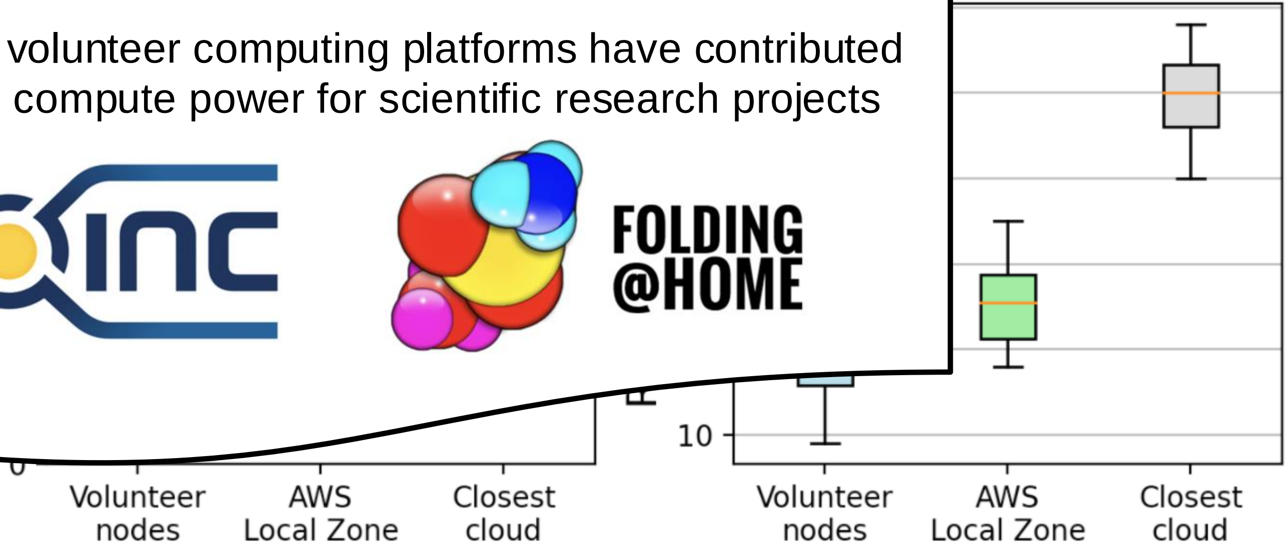
On-premise solution	Expensive to maintain private infrastructures/hardware
	Lack of scaling capacity

Can volunteer resources come into play?

Existing volunteer computing platforms have contributed massive compute power for scientific research projects

BOINC

FOLDING @HOME



(a) Network hops (b) RTT propagation latency

Network measurements in Minneapolis-Saint Paul metropolitan area

Can volunteer resources come into play?

They are widely/densely distributed with unlimited potential to scale cost-efficiently under appropriate incentive models

Volunteer resources are **greener complements** of existing edge infrastructures to enable **elastic edge computing** everywhere

They are powerful: personal PCs/laptops/devices are equipped with faster cpu/gpu/storage hardware

Now challenges…

Heterogeneous client-to-edge networks

Heterogeneous edge nodes

Dense and geo-distributed resource distribution

Unreliable edge node

→ ***Heterogeneous edge-dense environments***

Our objective

Achieve **edge elasticity** in ***Heterogeneous edge-dense environments***

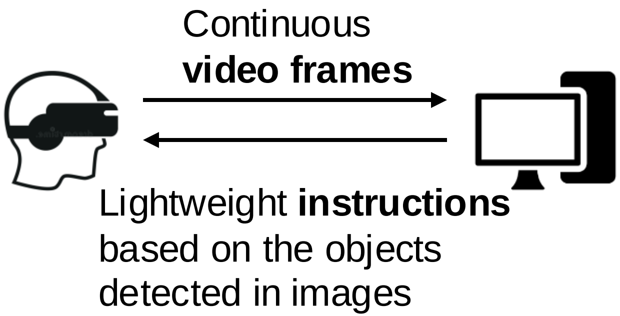
Specifically…

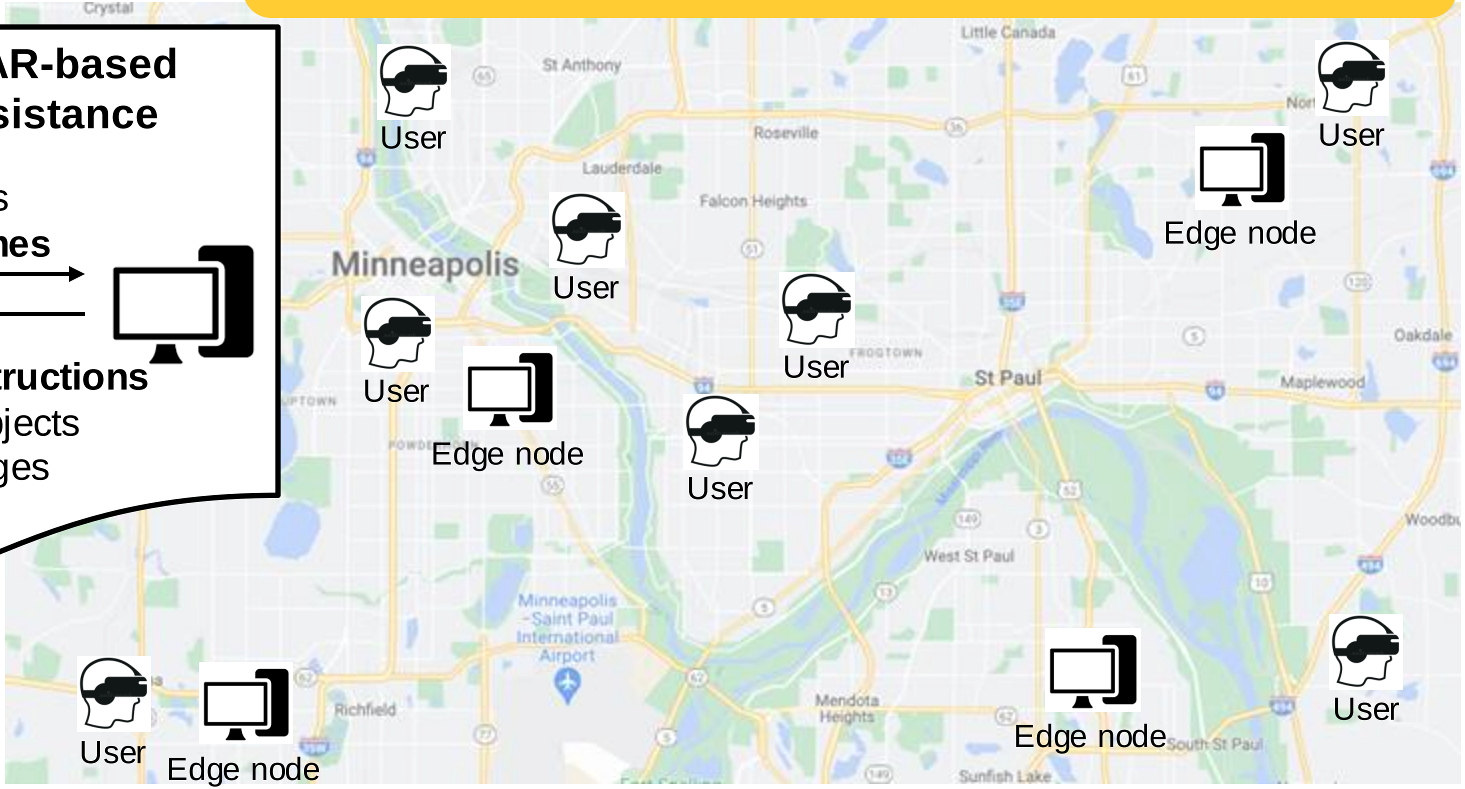
In a system with ***n* users** and ***m* edge nodes**, how to minimize the **average end-to-end latency** perceived by all users in *Heterogeneous edge-dense environments*

An example

Find the optimal edge node for each user such that global average latency performance is minimized

Application: AR-based cognitive assistance





Problem formulation

Consider a *heterogeneous edge-dense environment* with **n users** and **m edge nodes** in a specified area.

Edge Assignment (EA): A **users-to-edge match** that assigns each user u_i $(1 \leq i \leq n)$ an edge node e_j $(1 \leq j \leq m)$ to offload computation.

$$EA = \{< u_1, e_{j_1} >, < u_2, e_{j_2} >, \ldots, < u_n, e_{j_n} >\} \overset{\text{equivalent}}{\Leftrightarrow} EA = \{S_1, S_2, \ldots, S_m\}$$

From edge node's view: $\boldsymbol{S_j}$ denotes the **set of users** attached to edge node $\boldsymbol{e_j}$

Objective function:

End-to-end latency

$$P(EA) = \frac{1}{n}\sum_{i=1}^{n}\left(D_{prop}{}_i^{j_i} + D_{trans}{}_i^{j_i} + D_{proc}(e_{j_i}, S_{j_i})\right)$$

$$\underset{EA \in \Phi}{Min}\ P(EA)$$

Propa

dela

There are totally $\Phi = m^n$ possible EAs.

Queuing + Processing delay: determined by i) **node capacity** $\boldsymbol{e_{j_i}}$ and ii) **existing workload** $\boldsymbol{S_{j_i}}$ on this node

Problem formulation

$$\underset{EA\in\Phi}{Min}\, P(EA) = \frac{1}{n}\sum_{i=1}^{n}\left(D_{prop}{}_i^{j_i} + D_{trans}{}_i^{j_i} + D_{proc}(e_{j_i}, S_{j_i})\right)$$

D_{prop} and D_{trans} are only subject to client-centric views

- Client-centric (distributed) edge selection approach

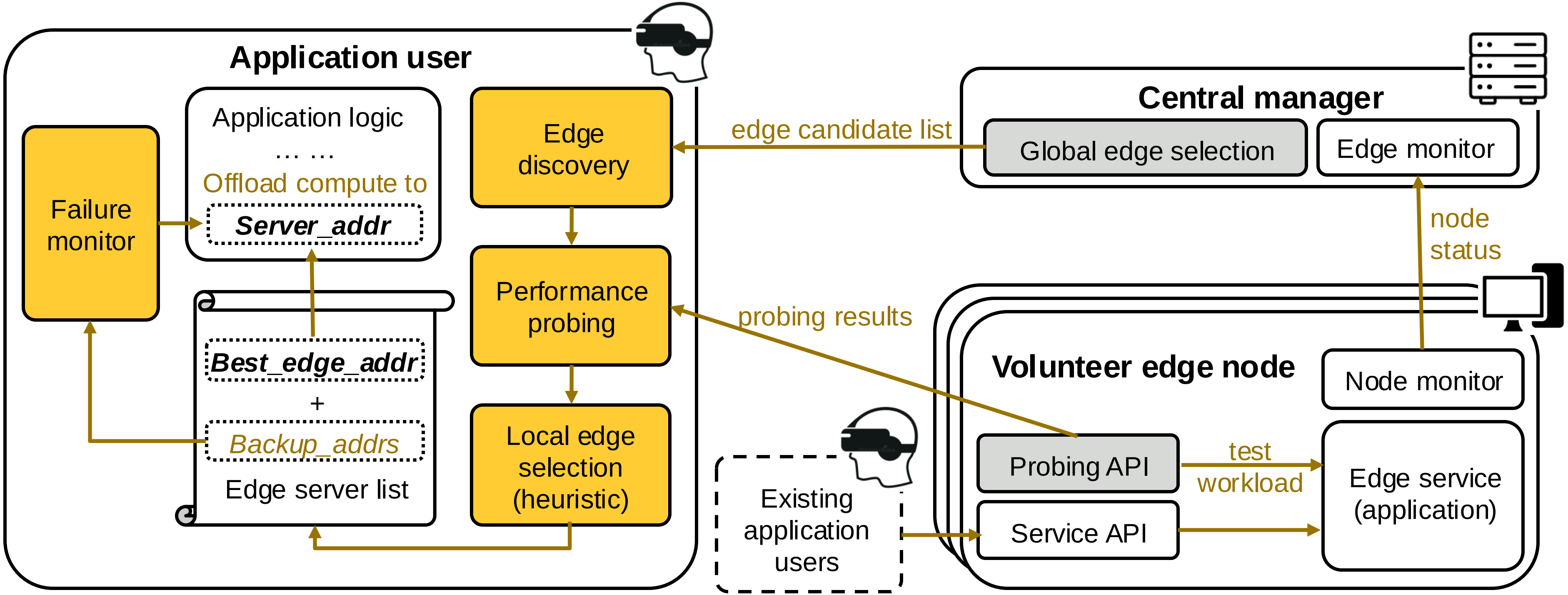
D_{proc} is varying under different hardware and resource contention levels

- Lightweight and accurate performance profiling process

Both users and edge nodes are dynamic with high node churn

- Adapt to system dynamics in real-time
- Fault tolerance mechanisms to guarantee continuous services

System design



1. Edge discovery

For each user who wants to discover nearby edge resources, we employ a **2-step approach**: (1) **Global edge selection** followed by a (2) **Local edge selection**.

Global edge selection: Central manager examines present edge nodes on **certain factors** to generate a coarse-grained ***Candidate edge list***.

- Geo-proximity
- Resource utilization
- Network affiliation
- Customized tags

Candidate edge list: A subset of edge nodes that are **expected** to provide low latency responses for specific users

TopN: size of the Candidate edge list

- ***TopN*** is an important configurable system parameter in our design
- Larger ***TopN*** value brings higher accuracy and flexibility to the edge selection process, but also introduce higher overhead

2. Performance probing

After the user obtains the ***Candidate edge list*** (with ***TopN*** candidates in the list), it applies a **probing approach** to predict edge performance during runtime.

Performance probing: Initiated by end-users directly to ***TopN*** candidate edge nodes to collect (1) **end-to-end networking metrics**, and (2) **“what-if” processing performance**.

- D_{prop}: **RTT propagation delay** from the user to the testing candidate edge node
- D_{trans}: **Data transfer delay** limited by the available bandwidth between the user and the testing candidate edge node

Easy to test by Ping

Consume currently available bandwidth and **compete** existing networking traffic

2. Performance probing

After the user obtains the ***Candidate edge list*** (with ***TopN*** candidates in the list), it applies a **probing approach** to predict edge performance during runtime.

Perf es to
colle **ance**.

Probing result = propagation delay + “what-if” processing time

“what-if” performance: the processing time measured by invoking a ***test synthetic workload*** to simulate “new-user-join” scenarios.

Test workload: Synthetic workload (compute offload request) based on the same application logic and compute requirements as the real offloading task

3. Local edge selection

After the user has the probing results of all edge candidates, local edge selection policy is used to sort the ***Candidate edge list*** to identify the best candidate.

Loc

sma

Local edge selection heuristic: $BLC = \min_{j=1}^{TopN} GO_j$

Global−view Overhead ($\boldsymbol{GO_j}$): considering the interference to existing workload on edge node j.

$$GO_j = \underline{n_j} \times (\underline{D\_proc_{probing} - D\_proc_{current}}) + LO_j$$

#existing users on node j

Performance degradation

Evaluation

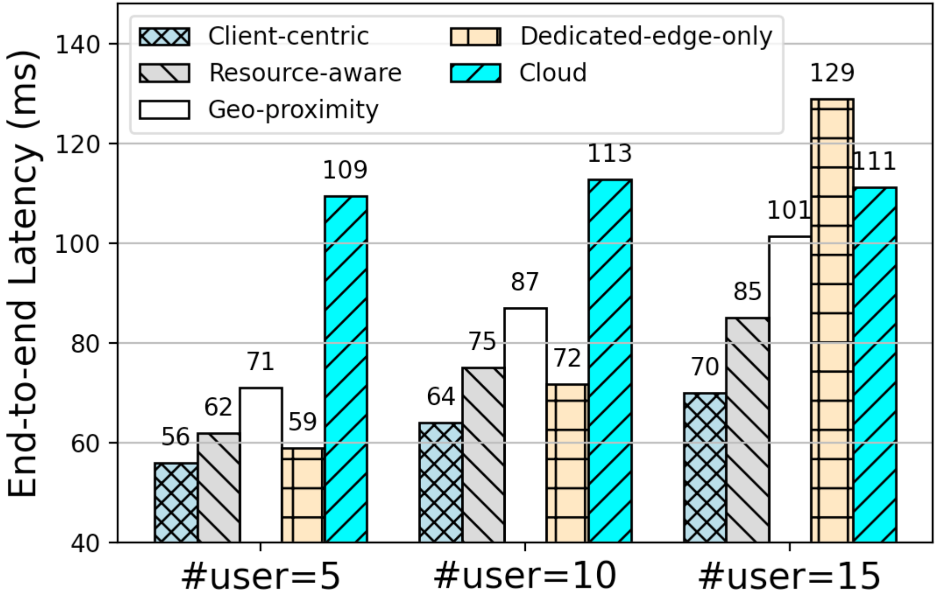
Real-world experiments setup:

- 20 participants in 10-mile radius, Minneapolis-Saint Paul metro area
- 15 users, 5 volunteer edge nodes, 4 AWS Local Zone

Node	Processor	Processing time – single video frame (ms)
V1	IntelR Core™ i7-9700, 8 cores	24
V2	IntelR Core™ i7-2720, 6 cores	32
V3	IntelR Core™ i9-8950HK, 6 cores	31
V4	IntelR Core™ i5-8250U, 4 cores	45
V5	IntelR Core™ i5-5250U, 2 cores	49
D6-D9	AWS Local Zone t3.xlarge	30
Cloud	AWS ec2 t3.xlarge	30

Edge Elasticity

Baselines: Geo-proximity, resource-aware weighted round-robin, dedicated-edge-only, closest cloud



✓Client-centric approach scales with load

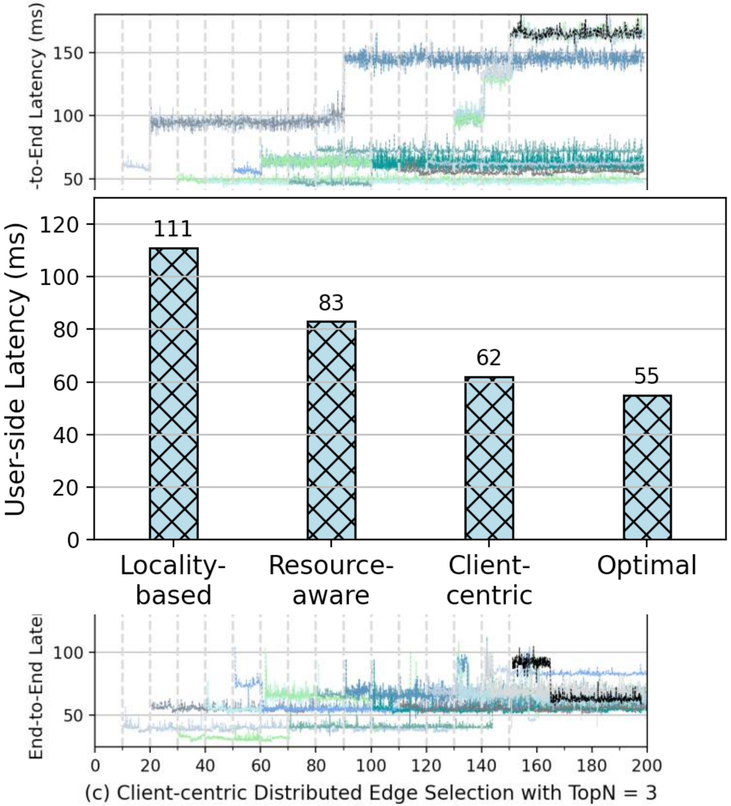
Static edge with increasing #users

- 9 volunteer nodes (4 x t2.medium, 4 x t2.xlarge, 1 x t2.2xlarge), 15 application users (15 x t2.micro)
- Within 50 miles, RTT ϵ [8, 55] ms

(a) Resource contention leads to overloading of local nodes

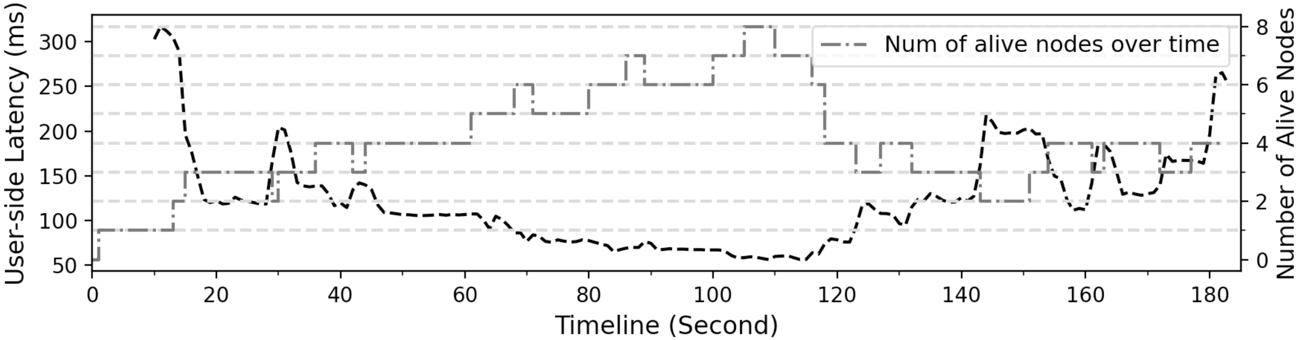
(b) Inability to identify network heterogeneity

(c) Performance probing and multi-node connection lead to low latency



Static users with high edge churn

- Edge node arrivals – Poisson distribution
- Edge node lifetime – Weibull distribution
- 18 edge nodes (8 x t2.medium, 8 x t2.xlarge, 2 x t2.2large)



✓ Correlation between average performance and edge resource availability

✓ Effective load balancing leads to low latency when new edge nodes join

Conclusion

- Existing edge deployments are not sufficient to support elastic edge computing everywhere. **Volunteer resources** can be greener compliments to existing edge infrastructures.
- We present the notion of ***Heterogeneous edge-dense environments***, and formulate a latency optimization problem towards edge elasticity.
- We design and implement a **client-centric edge selection approach** to achieve a near-optimal performance in dynamic environments.

UNIVERSITY OF MINNESOTA

Driven to Discover®

Crookston Duluth Morris Rochester Twin Cities